05-Scapy introduction

Security questions

Step 1

```
python#!/usr/bin/env python3
from scapy.all import*
# Creating two new objects, IP and ICMP.
# IP - is an object of an IP header, setting it's src and dst IP's.
# ICMP - is an object of an ICMP header, setting it's type to 8 (request).
a=IP(src="1.2.3.4", dst="10.0.2.4") / ICMP(type=8, code=0)
send(a)
```

- what is the purpose of this scapy script? This file sends a ICMP Echo request with a different SRC, a normal PING
- please visit the scapy manual page: https://scapy.readthedocs.io/en/latest/

Step 2

```
#!/usr/bin/env python3
from scapy.all import*
# Variable i is a counter of the number of routers.
i=1
# Sending a request with ttl i to the destination - 8.8.8.8
ans=sr1(IP(dst='8.8.8.8',ttl=i)/ICMP(),verbose=0, timeout=1)
print(i,""+ans.src)
# Loop while the packet recv isn't from our destination
while ans.src!= '8.8.8.8':
    # Incrementing the ttl time by 1.
    i+=1
    # Re-sending the ping request with incremented ttl.
    ans_temp=sr1(IP(dst='8.8.8.8',ttl=i)/ICMP(),verbose=0, timeout=1)
    # Checking for an unresponding router.
    if ans_temp is None:
        print(i,"******")
        continue
    else:
        ans=ans_temp
        # Printing router src address.
        print(i,""+ans.src)
```

- do you remember how traceroute works and what it does?

Traceroute uses the TTL for identifying the routers on the route by increment for each no request. The router decrement the TTL and if the TTL reach 0 the router sends back a ICMP TTL Exceed with the routers ip. This way traceroute can get the route

Step 3

```
#!/usr/bin/env python3
from scapy.all import*
def print_pkt(pkt):
# Printing the packet
    pkt.show()
# Calling a sniffing function from scapy library.
# iface is the interface the sniff should listen to.
# filter is the filter expression which is compiled to bpf.
# prn is a pointer to a function the packet that is sniffed is sent to.
pkt = sniff(iface=['eth0'], filter='icmp', prn=print_pkt)
```

- can you see the ftp credentials? yes

NMAP

```
#! /usr/bin/python
from scapy.all import *
load_module("nmap")
nmap_fp("192.168.1.1")
```

Step 4

```
#!/usr/bin/env python3
from scapy.all import*

source = SniffSource(iface=conf.iface)
wire = WiresharkSink()
def transf(pkt):
    if not pkt or IP not in pkt:
        return pkt
    pkt[IP].src = "1.1.1.1"
    pkt[IP].dst = "2.2.2.2"
    return pkt

source > TransformDrain(transf) > wire
p = PipeEngine(source)
p.start()
p.wait_and_stop()
```

- what could be the benefit of this script? This could be helpful to post wireshark output to a forum.

- it is possible to change information in a given pcap file too.
- please run a quick google search and find out how you could anonymize packages in a pcap file
- add this information to your write-up

```
#! /usr/bin/env python3
from scapy.utils import rdpcap, wrpcap
from scapy.layers.inet import IP, TCP, UDP  # import needed!
import sys

INFILE = "ipp.pcap"
OUTFILE = "ipp_an.pcap"

packets = rdpcap(INFILE)

new_packets = []
for packet in packets:
    if packet.haslayer("IP"):
        packet[IP].src = "1.1.1.1"
        packet[IP].dst = "2.2.2.2"

    new_packets.append(packet)

wrpcap(OUTFILE, new_packets)
```