Category:

network

Name:

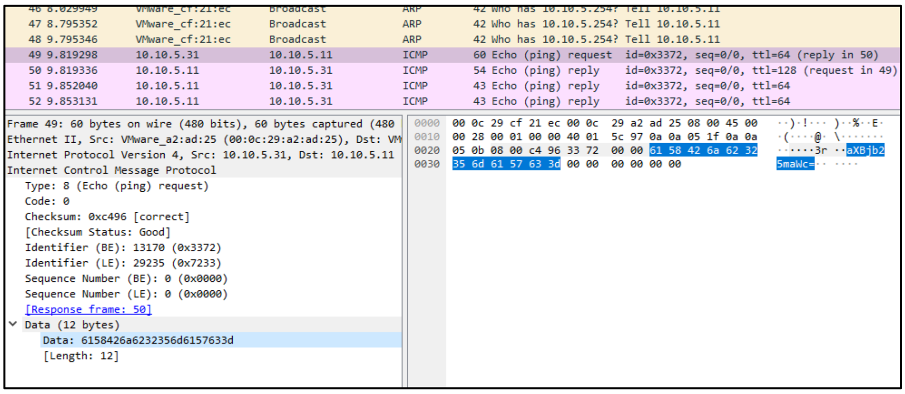
icmp

Message:

analyze the pcap file and find the flag.

Instructions:

Open the pcapng file with wireshark and check the contents. As the title of the challenge suggests, you can check ICMP communication between two hosts. It also turns out that the data payload part contains different data than normal ICMP packets.

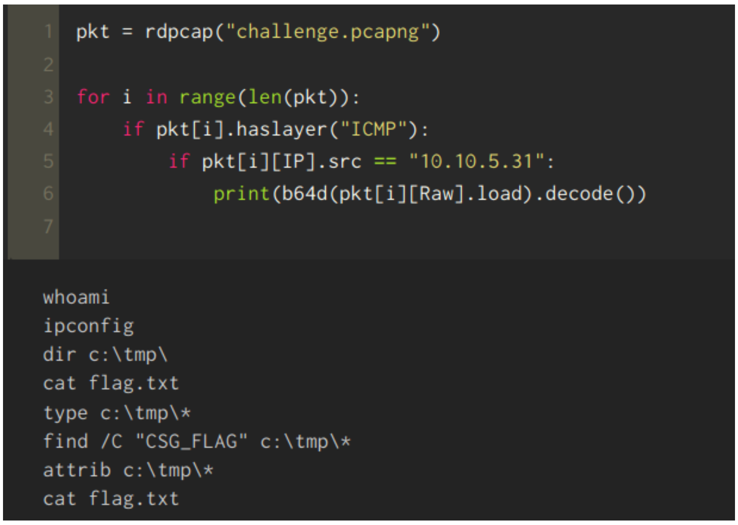


```
46 8.029949   VMware_cf:21:ec   Broadcast    ARP    42 Who has 10.10.5.254? Tell 10.10.5.11
47 8.795352   VMware_cf:21:ec   Broadcast    ARP    42 Who has 10.10.5.254? Tell 10.10.5.11
48 9.795346   VMware_cf:21:ec   Broadcast    ARP    42 Who has 10.10.5.254? Tell 10.10.5.11
49 9.819298   10.10.5.31        10.10.5.11   ICMP   60 Echo (ping) request  id=0x3372, seq=0/0, ttl=64 (reply in 50)
50 9.819336   10.10.5.11        10.10.5.31   ICMP   54 Echo (ping) reply    id=0x3372, seq=0/0, ttl=128 (request in 49)
51 9.852040   10.10.5.11        10.10.5.31   ICMP   43 Echo (ping) reply    id=0x3372, seq=0/0, ttl=64
52 9.853131   10.10.5.11        10.10.5.31   ICMP   43 Echo (ping) reply    id=0x3372, seq=0/0, ttl=64
```

```
Frame 49: 60 bytes on wire (480 bits), 60 bytes captured (480
Ethernet II, Src: VMware_a2:ad:25 (00:0c:29:a2:ad:25), Dst: VM
Internet Protocol Version 4, Src: 10.10.5.31, Dst: 10.10.5.11
Internet Control Message Protocol
  Type: 8 (Echo (ping) request)
  Code: 0
  Checksum: 0xc496 [correct]
  [Checksum Status: Good]
  Identifier (BE): 13170 (0x3372)
  Identifier (LE): 29235 (0x7233)
  Sequence Number (BE): 0 (0x0000)
  Sequence Number (LE): 0 (0x0000)
  [Response frame: 50]
v Data (12 bytes)
    Data: 6158426a6232356d6157633d
    [Length: 12]
```

```
0000  00 0c 29 cf 21 ec 00 0c  29 a2 ad 25 08 00 45 00   ··)·!··· )··%··E·
0010  00 28 00 01 00 00 40 01  5c 97 0a 0a 05 1f 0a 0a   ·(····@· \·······
0020  05 0b 08 00 c4 96 33 72  00 00 61 58 42 6a 62 32   ······3r ··aXBjb2
0030  35 6d 61 57 63 3d 00 00  00 00 00 00               5maWc=·· ····
```

From its data format, it is clear that the payload data is BASE64 encoded. ICMP reply packets also contain the same data. The first step is to collect these data and attempt BASE64 decoding. You may find some commands.



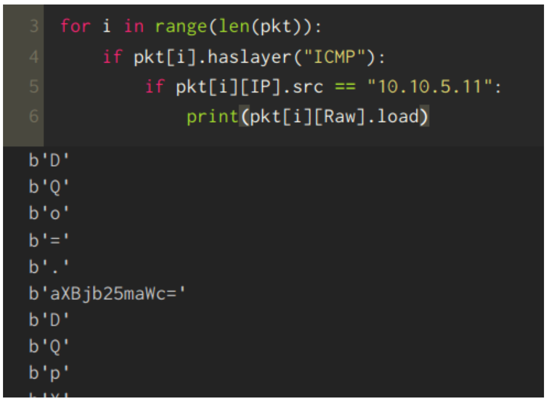
```python
pkt = rdpcap("challenge.pcapng")

for i in range(len(pkt)):
    if pkt[i].haslayer("ICMP"):
        if pkt[i][IP].src == "10.10.5.31":
            print(b64d(pkt[i][Raw].load).decode())

```

```
whoami
ipconfig
dir c:\tmp\
cat flag.txt
type c:\tmp\*
find /C "CSG_FLAG" c:\tmp\*
attrib c:\tmp\*
cat flag.txt
```

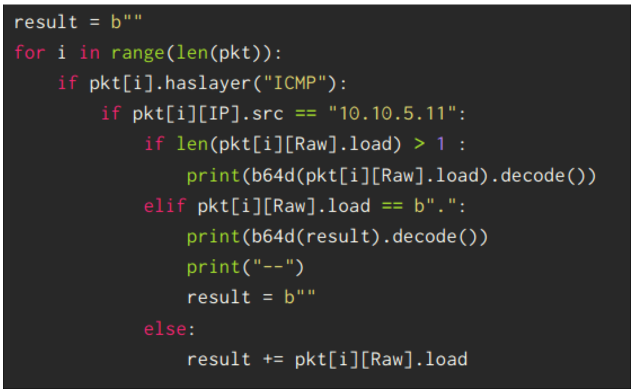
Next, check the response packet. In addition to the command confirmed in the previous step, you can see that it responds with data 1 byte at a time.



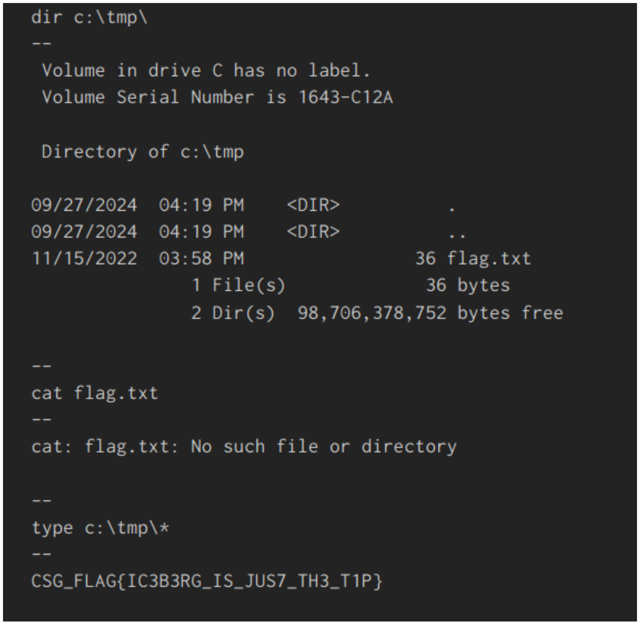
```
for i in range(len(pkt)):
    if pkt[i].haslayer("ICMP"):
        if pkt[i][IP].src == "10.10.5.11":
            print(pkt[i][Raw].load)
```

```
b'D'
b'Q'
b'o'
b'='
b'.'
b'aXBjb25maWc='
b'D'
b'Q'
b'p'
```

Since dot (“.”) is a symbol that is not used in normal BASE64 and is only used in the last response packet, it is treated as a delimiter that indicates the end of response data. You can obtain FLAG by using the following script.



```
result = b""
for i in range(len(pkt)):
    if pkt[i].haslayer("ICMP"):
        if pkt[i][IP].src == "10.10.5.11":
            if len(pkt[i][Raw].load) > 1 :
                print(b64d(pkt[i][Raw].load).decode())
            elif pkt[i][Raw].load == b".":
                print(b64d(result).decode())
                print("--")
                result = b""
            else:
                result += pkt[i][Raw].load
```



```
dir c:\tmp\
--
 Volume in drive C has no label.
 Volume Serial Number is 1643-C12A

 Directory of c:\tmp

09/27/2024  04:19 PM    <DIR>          .
09/27/2024  04:19 PM    <DIR>          ..
11/15/2022  03:58 PM                36 flag.txt
               1 File(s)             36 bytes
               2 Dir(s)  98,706,378,752 bytes free

--
cat flag.txt
--
cat: flag.txt: No such file or directory

--
type c:\tmp\*
--
CSG_FLAG{IC3B3RG_IS_JUS7_TH3_T1P}
```