گزارشکار پروژه ی دوم درس شبکه های کامپیوتری

استاد: دکتر نادران طحان

اعضای تیم: زهرا معصومی (9873131) ، سارینا همت پور (9873136)

زبان برنامه نویسی استفاده شده، پایتون میباشد.

در این پروژه قصد داریم با خواندن تعدادی packet، اطلاعات مربوط به آن ها را نمایش دهیم. برای خواندن packetها دو راه داریم. راه اول، sniff کردن و راه دوم استخراج آن ها از فایل trace است که این ها با کتابخانه ی scapy امکان پذیر است. البته scapy به تنهایی جوابگو نیست زیرا تا لایه ی دوم کار میکند. برای اینکه بتوانیم کارایی را تا لایه ی سوم ارتقا دهیم نیاز به استفاده از کتابخانه ی npcap (یا winpcap) داریم. پس آن ها را import میکنیم.

```
1
2   from sys import flags
3   from telnetlib import IP
4   from scapy.all import *
5
```

برای شروع کار نیاز به یک سری پکت داریم. که راه اول sniff کردن بود.

```
6
7       a = sniff(count=4)
8       a.summary()
9
```

در تصویر بالا 4 پکت خوانده میشود و با تابع summary() خلاصه ای از اطلاعات مجموعه پکت ها نمایش داده میشود.

راه دوم استفاده از Trace table هاست. در ادامه نیز خروجی ها را بر اساس این راه نشان خواهیم داد.

```
10
11    packets = rdpcap("C:/Users/Lenovo/Downloads/Compressed/trace4.pcap")
12
```

- همانطور که در تصویر بالا مشخص است با تابع rdpcap() میتوان فایل trace را خواند. ورودی این تابع، آدرس فایل مورد نظر در حافظه ی محلی میباشد. خروجی این تابع نیز هم همان پکت ها هستند که در متغیر packets ذخیره شده اند.

```
21
22    #the whole summary
23    print(packets)
24
```

با پرینت کردن آن میتوان خلاصه ای از تعداد پکت ها بدست آوریم. خروجی آن هم به صورت زیر است. (نمایش تعداد پکت های TCP، UDP، ICMP و غیره)

```
<attack1.pcap: TCP:86650 UDP:30 ICMP:0 Other:6>
```

- برای بدست آوردن تعداد کل پکت ها از تابع زیر استفاده میکنیم.

```
25
26    #number off all packets
27    length=len(packets)
28    print(f"\nNumber of all packets: {length}")
29
```

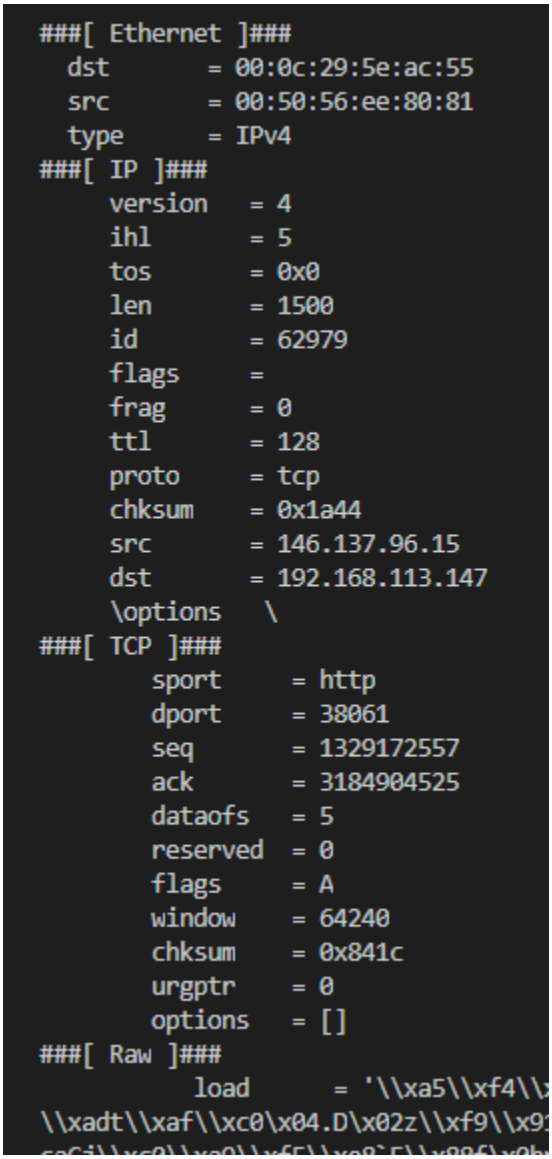
خروجی:

```
Number of all packets: 86686
```

- برای چاپ کردن هدر لایه های مختلف پکت ها از تابع show() در یک حلقه استفاده میکنیم.

```
14
15    #show fields:
16    for i in packets:
17        i.show()
18
```

نمونه خروجی برای یک پکت از نوع TCP:



- برای شمارش تعداد پکت های TCP و UDP و نسبت درصد آنها به کل بسته ها به یک حلقه و دو شرط نیاز داریم.

```
30
31  #percentage of tcp and udp packets
32  countTCP=0
33  countUDP=0
34
35  for i in range(0, len(packets)):
36      pkt = packets[i]
37      if (TCP in pkt):
38          countTCP+= 1
39      elif (UDP in pkt):
40          countUDP+= 1
41
42  print(f"\nTCP packets= {(countTCP/length)*100}%")
43  print(f"UDP packets= {(countUDP/length)*100}%")
44
```

در صورت درست بودن عبارت TCP in pkt یکی به تعداد پکت های TCP و در صورت درست بودن UDP in pkt نیز یکی به تعداد پکت های UDP اضافه میشود. در آخر هم نسبت درصد آنها محاسبه و چاپ میشود.

نمونه خروجی:

```
TCP packets= 99.95847080266 71%
UDP packets= 0.03460766444408555%
```

- پس از آن نیاز به به دست آوردن تعداد دیتاگرم های فرگمنت شده داریم. برای این کار از فیلدflags هدر IP استفاده می کنیم. این فیلد 3 بیت دارد که بیت اول آن 0 است.

اما بیت های دوم و سوم آن با توجه به مشخصات بسته می توانند 0 یا 1 باشند. بیت دوم DF مخفف don’t fragment است که درصورت 1 بودن، نشان می دهد که این دیتاگرم نباید فرگمنت شود.(بیت سوم هم MF است که در اینجا مورد بحث ما نیست.)

پس ما با داشتن تعداد دیتاگرم هایی که فرگمنت نمی شوند، و کم کردن آن از تعداد کل دیتاگرم ها می توانیم تعداد دیتاگرم های فرگمنت شده را به دست آوریم که مطابق تصویر محاسبه می شود:

```
48  DFs = 0
49  for i in packets[IP]:
50      if i.flags==2: #DF  010 i.flags=="DF"
51          DFs+=1
52
53  fragmented = length-DFs
54
55  print(f"\nNumber of fragmented datagrams= {fragmented}")
56
```

با گشتن در هدر IP تمام پکت ها، تعداد پکت هایی که فیلد flags آن ها DF است یا مقدار 2 دارد(بیت دوم آن مقدار 1 دارد) را می شماریم و در متغیر DFs ذخیره می کنیم.

در نهایت آن را از تعداد کل پکت ها که قبلا به دست آورده ایم کم می کنیم تا تعداد پکت های فرگمنت شده مشخص شود.

نمونه خروجی به صورت زیر است:

```
**********Number of fragmented datagrams:**********

Number of fragmented datagrams= 60082
```

- برای تشخیص حمله scanning در یک اتصال نیاز به بیت های SYN و SYN/ACK در هدر لایه4 داریم که در فیلد flags قراردارند. این فیلد 6 بیت دارد که بیت های دوم و پنجم آن مربوط به SYN و SYN/ACK می باشند.

باید نسبت تعداد پکت ها با بیت مثبت(1) SYN (درخواست) به تعداد پکت ها با هر دو بیت SYN و SYN/ACK (درخواست و تایید) با مقدار 1 به دست آوریم.

و در صورتی که این نسبت در یک اتصال از 1 به 3 بیشتر شد تشخیص حمله scanning دهیم.

برای در نظر گرفتن یک اتصال ابتدا آدرس IP از ورودی گرفته و با جستجو در فیلد flags هدر TCP بسته های مربوط به اتصال آدرس IP موردنظر، تعداد پکت ها با بیت مثبت(1) SYN و پکت ها با هر دو بیت SYN و SYN/ACK را می شماریم و نسبت می گیریم:

```
#scanning **************************
srcIP=input('\n\nEnter the source IP:  ')
countS=0
countSA = 0

for i in packets[TCP]:
    if i.src == srcIP:
        if i.flags == 18:
            countSA +=1

        if i.flags == "S":
            countS +=1


print(f"\nSYN/ACK: {countSA}")
print(f"SYN: {countS}")
```

مقدار فیلدی که بیت های اول و پنجم آن 1 باشند 18 است.(SYN = 1 + SYN/ACK = 1)

و فیلدی که بیت SYN آن 1 باشد مقدار S می گیرد.

از این طریق تعداد را محاسبه می کنیم.

حال نسبت را به دست می آوریم و در صورت تصدیق شرایط ذکر شده، تشخیص حمله می دهیم:

```
print(f"\nSYN/ACK: {countSA}")
print(f"SYN: {countS}")

if countSA!=0:
    if countS/countSA>3:
        print(f"\nWARNING _ scanning attack !!! {countS/countSA>3}")
else:
    print('\n no attack detected')
```