

امیررضا حسینی ۹۸۲۰۳۶۳

مهسا امینی ۹۸۱۷۸۲۳

پروژه دوم

قسمت اول: ارسال هر بسته دلخواه:

سوال (۱) حداقل طول برابر ۱۴ بایت هست. با آزمون و خطا و همچنین دیدن بسته های وایرشارک (زمانی که دیتا خالی باشد) و در ۶ بایت برای مبدا و ۶ بایت نیز برای مقصد و دو بایت هم برای مشخص کردن تایپ اجباری هست.

سوال (۲) بعد از capture کردن در wireshark متوجه میشویم که پروتکل استفاده شده از نوع زیر هست.

[Protocols in frame: eth:ethertype]

سوال (۳)

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	Shenzhen 22:fc:dc	Broadcast	ARP	60	Who has 192.168.0.183? Tell 192.168.0.254
2	0.222155371	12:34:56:78:90:12	98:02:30:00:00:21	0xbcac	14	Ethernet II
3	1.023524282	Shenzhen 22:fc:dc	Broadcast	ARP	60	Who has 192.168.0.183? Tell 192.168.0.254
4	1.690810379	192.168.0.180	172.20.11.11	DNS	113	Standard query 0xa9eb A onedscolprdwus11.westus.cloudapp.azure...
5	1.692645820	192.168.0.1	239.255.255.250	SSDP	307	NOTIFY * HTTP/1.1
6	1.696170765	192.168.0.1	239.255.255.250	SSDP	316	NOTIFY * HTTP/1.1
7	1.696171074	192.168.0.1	239.255.255.250	SSDP	379	NOTIFY * HTTP/1.1
8	1.696171121	192.168.0.1	239.255.255.250	SSDP	371	NOTIFY * HTTP/1.1
9	1.696171168	192.168.0.1	239.255.255.250	SSDP	316	NOTIFY * HTTP/1.1
10	1.696171216	192.168.0.1	239.255.255.250	SSDP	355	NOTIFY * HTTP/1.1
11	1.696171268	192.168.0.1	239.255.255.250	SSDP	387	NOTIFY * HTTP/1.1
12	1.699933636	192.168.0.1	239.255.255.250	SSDP	316	NOTIFY * HTTP/1.1
13	1.699934012	192.168.0.1	239.255.255.250	SSDP	375	NOTIFY * HTTP/1.1
14	1.699934066	192.168.0.1	239.255.255.250	SSDP	381	NOTIFY * HTTP/1.1
15	1.699934119	192.168.0.1	239.255.255.250	SSDP	369	NOTIFY * HTTP/1.1
16	1.699934171	192.168.0.1	239.255.255.250	SSDP	375	NOTIFY * HTTP/1.1
17	1.600034224	192.168.0.1	239.255.255.250	SSDP	305	NOTIFY * HTTP/1.1
Frame 2: 14 bytes on wire (112 bits), 14 bytes captured (112 bits) on interface enp0s3, id 0						
Ethernet II, Src: 12:34:56:78:90:12 (12:34:56:78:90:12), Dst: 98:02:30:00:00:21 (98:02:30:00:00:21)						
Destination: 98:02:30:00:00:21 (98:02:30:00:00:21)						
Source: 12:34:56:78:90:12 (12:34:56:78:90:12)						
Type: Unknown (0xbcac)						
0000 98 02 30 00 00 21 12 34 56 78 90 12 bc ac ..0...!..4 Vx...						
Type (eth.type), 2 bytes						

```
amir@amir-VirtualBox:~/Desktop/shabake project$ sudo python3 pkt_sender.py
What is your packet content?
980230000021123456789012bcac
Which interface do you want to use?
enp0s3
send 14-bytes packet on enp0s3
amir@amir-VirtualBox:~/Desktop/shabake project$
```

سوال ۴) در این سوال همانطور که میبینیم بسته ای به طول ۶۶ بایت ارسال شده که مشخصات آن دقیقاً با بسته اولیه معادل هست و از این رو میفهمیم همان بسته‌ای است که توسط برنامه ارسال شده.

tcp.port==4686						
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.0.115	8.8.8.8	TCP	66	4686 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_P...
12	0.063922369	8.8.8.8	192.168.0.115	TCP	66	443 → 4686 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1430 SA...
14	0.063922718	192.168.0.115	8.8.8.8	TCP	60	4686 → 443 [ACK] Seq=1 Ack=1 Win=65536 Len=0
16	0.085575148	192.168.0.115	8.8.8.8	TLSv1	571	Client Hello
18	0.379195960	192.168.0.115	8.8.8.8	TCP	571	[TCP Retransmission] 4686 → 443 [PSH, ACK] Seq=1 Ack=1 Win=65...
19	0.469097958	192.168.0.115	8.8.8.8	TCP	60	4686 → 443 [FIN, ACK] Seq=518 Ack=1 Win=65536 Len=0
24	0.680258621	192.168.0.115	8.8.8.8	TCP	571	[TCP Retransmission] 4686 → 443 [FIN, PSH, ACK] Seq=1 Ack=1 W...
35	1.280575691	192.168.0.115	8.8.8.8	TCP	571	[TCP Retransmission] 4686 → 443 [FIN, PSH, ACK] Seq=1 Ack=1 W...
75	2.481439425	192.168.0.115	8.8.8.8	TCP	571	[TCP Retransmission] 4686 → 443 [FIN, PSH, ACK] Seq=1 Ack=1 W...
108	4.882010761	192.168.0.115	8.8.8.8	TCP	571	[TCP Retransmission] 4686 → 443 [FIN, PSH, ACK] Seq=1 Ack=1 W...
229	9.682759043	192.168.0.115	8.8.8.8	TCP	571	[TCP Retransmission] 4686 → 443 [FIN, PSH, ACK] Seq=1 Ack=1 W...
378	16.033847364	192.168.0.115	8.8.8.8	TCP	66	[TCP Retransmission] 4686 → 443 [SYN] Seq=0 Win=64240 Len=0 M...

```

Frame 378: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface enp0s3, id 0
Ethernet II, Src: a4:6b:b6:ee:a4:9e (a4:6b:b6:ee:a4:9e), Dst: Shenzhen_22:fc:dc (d8:d8:66:22:fc:dc)
  Destination: Shenzhen_22:fc:dc (d8:d8:66:22:fc:dc)
  Source: a4:6b:b6:ee:a4:9e (a4:6b:b6:ee:a4:9e)
Type: IPv4 (0x0800)
  Internet Protocol Version 4, Src: 192.168.0.115, Dst: 8.8.8.8
  Transmission Control Protocol, Src Port: 4686, Dst Port: 443, Seq: 0, Len: 0

```

```

amir@amir-VirtualBox:~/Desktop/shabake project$ sudo python3 pkt_sender.py
What is your packet content?
d8d86622fcdca46bb6eea49e080045000034f6f24000400672a6c0a8007308080808124e01bb4e23cdc3000000008002faf073040000020405b40103030801010402
Which interface do you want to use?
enp0s3
send 66-bytes packet on enp0s3
amir@amir-VirtualBox:~/Desktop/shabake project$

```

سوال ۵) این نوع حمله از دسته حملاتی هست که در آن یک انتقال داده معتبر با انگیزه بدخواهانه یا کلاه برداری تکرار می‌شود یا به تاخیر می‌افتد.

از نوع حملات man-in-the-middle attack می‌باشد. مثلاً می‌توان در spoofing از آن استفاده کرد.

به این صورت می‌توانیم از برنامه pkt_sender.py به این صورت استفاده کرد که به بسته‌های مبادله شده به عنوان واسطه گوش دهیم و پس از دریافت آنها و استراق سمع اطلاعات، آنها را مجدد برای هدف ارسال نماییم و آدرس مقصد را آدرس اولیه بگذاریم. و با انجام این کار از دید دو طرف مخفی می‌مانیم.

تقریباً مانند کاری که در سوال چهارم و سوم انجام دادیم.

قسمت دوم: ارسال بسته های TCP

سوال (۱) بسته هایی که در رده دوم اهمیت برای سرور قرار میگیرند مثل ack number و up و یا حتی برخی از flag ها به شرط اینکه مفهوم بسته ارسالی تغییر نکند.

سوال (۲)

The image shows a Wireshark packet capture window titled '*enp0s3'. The filter bar shows 'ip.addr == 93.184.216.34'. The packet list shows three packets:

No.	Time	Source	Destination	Protocol	Length	Info
4	1.256559905	192.168.0.180	93.184.216.34	TCP	54	3984 → 80 [SYN] Seq=0 Win=4210 Len=0
6	1.539174311	93.184.216.34	192.168.0.180	TCP	60	80 → 3984 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1
7	1.539255613	192.168.0.180	93.184.216.34	TCP	54	3984 → 80 [RST] Seq=1 Win=0 Len=0

The status bar at the bottom shows: 'wireshark_enp0s3_20220627133806_qkLSA5.pcapng', 'Packets: 19 · Displayed: 3 (15.8%) · Dropped: 0 (0.0%)', and 'Profile: Default'.

سوال (۳) بسته های دیگر در واقع syn ack و ack هستند که برای handshaking اولیه ارتباط TCP ایجاد میشوند. و همانطور که مشخص هست دستگاه ما به طور مشخص در جواب دوم ارتباط را reset کرده است به دلیل اینکه برنامه ما به اتمام رسیده است و نمیخواهیم به ارتباط ادامه دهیم. از روی پورت مبدا و مقصد هم میتوان فهمید بسته از کدام سمت ارسال شده.

سوال ۴) نمونه خروجی اجرا شده توسط اسکریپت

```
GNU nano 4.8 ifinfo.sh
#!/bin/bash

ip_in=$1
server_port=$2
device=$3
client_port=$4

ip4=$(/sbin/ip -o -4 addr list $device | awk '{print $4}' | cut -d/ -f1)
read MAC </sys/class/net/$device/address
default_mac=$(arp -n | grep `route -n | awk '/UG/{print $2}` | awk '{print $3}')

echo $ip_in > infotest.txt
echo $server_port >> infotest.txt
echo $ip4 >> infotest.txt
echo $client_port >> infotest.txt
echo $device >> infotest.txt
echo $MAC >> infotest.txt
echo $default_mac >> infotest.txt
```

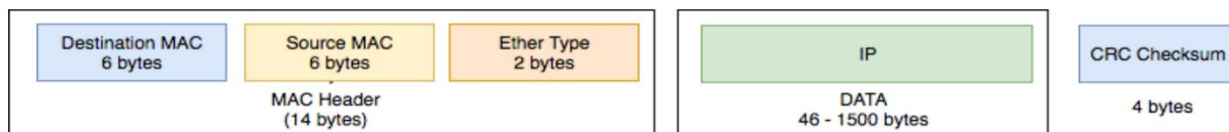
```
infotest.txt
1 93.184.216.34
2 80
3 192.168.0.180
4 3000
5 enp0s3
6 08:00:27:a2:20:b3
7 d8:d8:66:22:fc:dc
8
```

البتّه باید دسترسی `chmod 777` را برای اجرا به اسکریپت فوق بدهیم.

قسمت سوم: مینی-وایر شارک

سوال یک)

The Ethernet frame structure is as follows:



طول هدر اترنت همانطور که در سوال اول قسمت اول اشاره شد حداقل 14 بایت است.

سوال دو)

تابعی به نام tcp_head داریم که در آن flag_syn و flag_ack را بدست می آوریم اگر مقدار این دو یک باشد بسته های دریافتی ACK-SYN هستند.

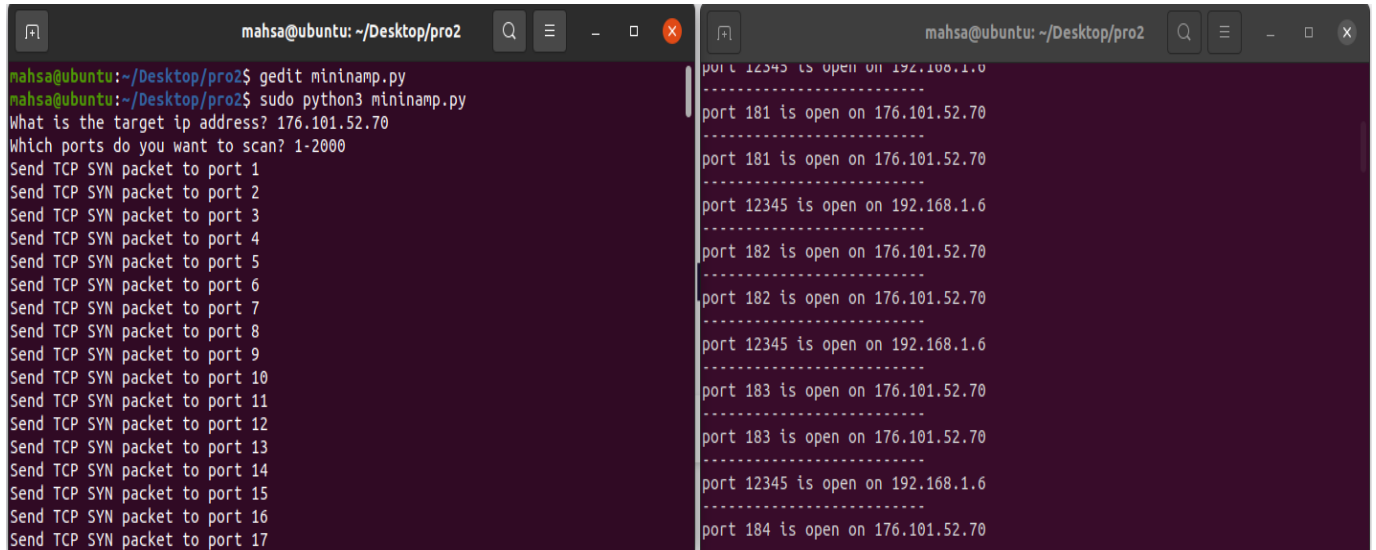
اجرا:

```
mahsa@ubuntu: ~/Desktop/pro2
mahsa@ubuntu:~/Desktop/pro2$ sudo python3 miniwiresark.py
port 80 is open on 192.168.0.180
-----
mahsa@ubuntu:~/Desktop/pro2$ sudo python3 pkt_sender.py
What is your packet content?
d8d86622fcdc080027a220b308004500002807c3000040067bd6c0a800b45db8d8220f9000501749
30d10000000050021072503f0000
Which interface do you want to use?
ens33
send 54-bytes packet on ens33
mahsa@ubuntu:~/Desktop/pro2$
```

```
mahsa@ubuntu:~/Desktop/pro2$ sudo python3 miniwiresark.py
port 443 is open on 192.168.1.6
-----
port 443 is open on 192.168.1.6
-----
port 80 is open on 192.168.1.6
-----
port 44418 is open on 34.107.221.82
-----
port 80 is open on 192.168.1.6
-----
port 80 is open on 192.168.1.6
-----
port 44418 is open on 34.107.221.82
-----
port 44418 is open on 34.107.221.82
-----
port 80 is open on 192.168.1.6
-----
port 80 is open on 192.168.1.6
-----
```

قسمت چهارم: مینی آن مپ

```
mahsa@ubuntu: ~/Desktop/pro2$ sudo python3 mininamp.py
What is the target ip address? 176.101.52.70
Which ports do you want to scan? 1-2000
Send TCP SYN packet to port 1
Send TCP SYN packet to port 2
Send TCP SYN packet to port 3
Send TCP SYN packet to port 4
```



```
mahsa@ubuntu: ~/Desktop/pro2$ gedit mininamp.py
mahsa@ubuntu: ~/Desktop/pro2$ sudo python3 mininamp.py
What is the target ip address? 176.101.52.70
Which ports do you want to scan? 1-2000
Send TCP SYN packet to port 1
Send TCP SYN packet to port 2
Send TCP SYN packet to port 3
Send TCP SYN packet to port 4
Send TCP SYN packet to port 5
Send TCP SYN packet to port 6
Send TCP SYN packet to port 7
Send TCP SYN packet to port 8
Send TCP SYN packet to port 9
Send TCP SYN packet to port 10
Send TCP SYN packet to port 11
Send TCP SYN packet to port 12
Send TCP SYN packet to port 13
Send TCP SYN packet to port 14
Send TCP SYN packet to port 15
Send TCP SYN packet to port 16
Send TCP SYN packet to port 17

port 12345 is open on 192.168.1.6
-----
port 181 is open on 176.101.52.70
-----
port 181 is open on 176.101.52.70
-----
port 12345 is open on 192.168.1.6
-----
port 182 is open on 176.101.52.70
-----
port 182 is open on 176.101.52.70
-----
port 12345 is open on 192.168.1.6
-----
port 183 is open on 176.101.52.70
-----
port 183 is open on 176.101.52.70
-----
port 12345 is open on 192.168.1.6
-----
port 184 is open on 176.101.52.70
```