گزارش پروژه‌ی دوم درس شبکه‌های کامپیوتری

امیرحسین رجب‌پور ۹۷۳۱۰۸۵

سوال ۱: تلنت (Telnet) یک پروتکل شبکه می‌باشد (مخفف Telecommunications Network) که برای دسترسی به کامپیوتر و برقراری تماس مبتنی بر متن استفاده می‌شود و در واقع کانال ارتباطی بین دو کامپیوتر می‌باشد. این پروتکل به منظور شکل‌دهی و مدیریت از راه دور کامپیوترها از ترمینال‌های مختلف بود. این سرویس به دو صورت سرویس دهنده و سرویس گیرنده می‌باشد. این پروتکل قدیمی می‌باشد و قدمت آن به سال ۱۹۶۹ برمی‌گردد. این پروتکل از برای انتقال از TCP و پورت 23 استفاده می‌کند.

این پروتکل یک پروتکل امن نمی‌باشد و کد کاربری و رمز عبور آن بدون رمزنگاری قابل نمایش است به همین دلیل (امنیت پایین) پروتکل SSH جایگزین آن شده‌است.

سوال ۲: در ابتدای توسعه و استفاده‌ی تلنت بحث امنیت خیلی مدنظر نبود اما در سال ۱۹۹۰ که انفجار پهنای باند رخ داد و دسترسی عموم مردم به اینترنت زیاد شد، نفوذ و سوءاستفاده زیاد شد درنتیجه بحث امنیت مهم شد. پروتکل تلنت نیز از رمزنگاری پیام‌ها استفاده نمی‌کند درنتیجه امنیت پایینی دارد و می‌توان پکت‌ها را شنود کرد و اطلاعات مهم را استخراج کرد. پروتکل جایگزین تلنت، SSH می‌باشد که رمزنگاری را به پیام‌ها اضافه کرد در نتیجه ارتباط امنی می‌تواند ایجاد کند.

سوال ۳: پروتکل TLS (Transport Layer Security) یکی از پروتکل‌هایی است که برای برقراری ارتباط‌های امن‌تر در شبکه (یک پروتکل کدگذاری است) طراحی شده‌است. درواقع این پروتکل نسخه‌ی جدیدتر و بهبود یافته‌تر پروتکل SSL می‌باشد (و بر پایه‌ی آن است). شیوه‌ی کار آن بدین صورت است که برای برقراری ارتباط بین کلاینت-سرور ابتدا باید مشخص باشد که کلاینت می‌خواهد از پروتکل TLS استفاده کند (به عنوان مثال با کمک یک پورت مشخص). بعد از آن فرآیند TLS handshake انجام می‌شود و در طی این فرآیند دو طرف ارتباط به کمک رمزنگاری نامتقارن، بر سر یک کلید مشترک توافق می‌کنند و سپس داده‌ها با بین کلاینت و سرور با استفاده از این کلید مشترک (داده در مبداء رمزنگاری و در مقصد رمزگشایی می‌شوند) جابجا می‌شوند.

خروجی بخش اول:

در صورتی که حجم دیتا (به عنوان مثال یک فایل) از مقدار PACKET_SIZE بیشتر باشد نیز به صورت صحیح دیتا منتقل می‌شود.

ارسال درخواست به یک وبسایت:

```
PS E:\uni\semester 6\Network\Projects\project 2\CN_Proj2_9731085> python client.py www.tcal.ir 80
host is www.tcal.ir, port is 80
connected to  193.141.64.60
>> GET / HTTP/1.1\r\nHost: tcal.ir\n\n
received from server:  HTTP/1.1 400 Bad Request
Content-Type: text/html; charset=us-ascii
Server: Microsoft-HTTPAPI/2.0
Date: Tue, 25 May 2021 09:23:19 GMT
Connection: close
Content-Length: 311

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN""http://www.w3.org/TR/html4/strict.dtd">
<HTML><HEAD><TITLE>Bad Request</TITLE>
<META HTTP-EQUIV="Content-Type" Content="text/html; charset=us-ascii"></HEAD>
<BODY><h2>Bad Request</h2>
<hr><p>HTTP Error 400. The request is badly formed.</p>
</BODY></HTML>

>>
```

نشان دادن پورت‌های باز یک هاست:

```
PS E:\uni\semester 6\Network\Projects\project 2\CN_Proj2_9731085> C:\Users\Asus\AppData\Local\Programs\Python\Python38\python.exe client.py find-open-ports
Host: www.google.com
minimum port range: 50
maximum port range: 100
open ports for host: www.google.com are: [80]
```

ایمیل:

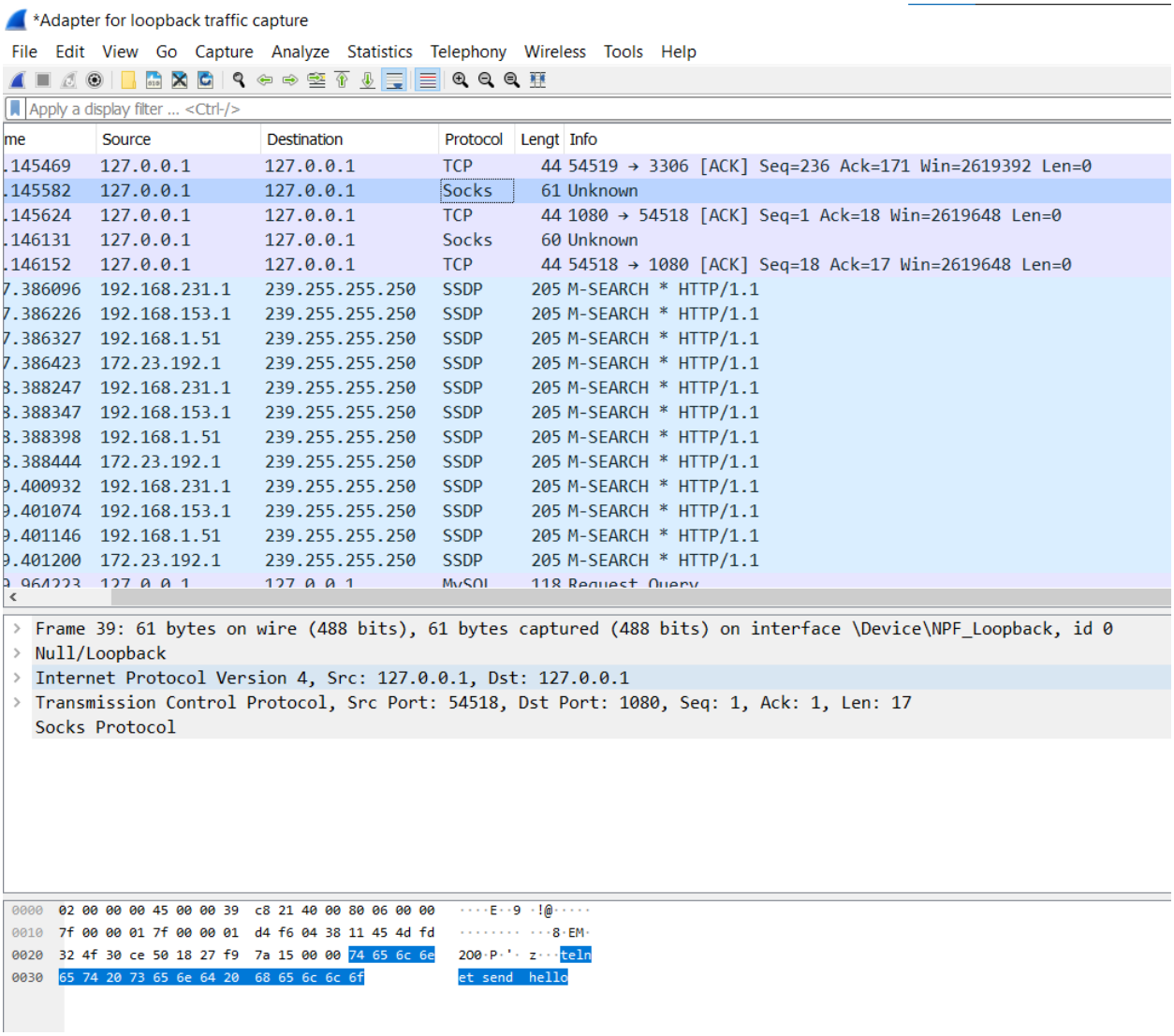
```
PS E:\uni\semester 6\Network\Projects\project 2\CN_Proj2_9731085> C:\Users\Asus\AppData\Local\Programs\Python\Python38\python.exe client.py aut.ac.ir 25
host is aut.ac.ir, port is 25
connected to  185.211.88.131
>> HELO aut.ac.ir
received from server:  220 asg.aut.ac.ir ESMTP ready.

>> MAIL FROM: test@test
received from server:  250 asg.aut.ac.ir Hello aut.ac.ir [86.55.170.180]

>> RCPT TO: rajabpoura@aut.ac.ir
received from server:  250 OK

>> data
received from server:  250 Accepted

>> this is a test mail from telnet project.
received from server:  354 Enter message, ending with "." on a line by itself

>> .
received from server:  250 OK id=1lmie2-0008T0-0G
```

نمونه‌ای از ارسال درخواست به سرور:

```
PS E:\uni\semester 6\Network\Projects\project 2\CN_Proj2_9731085> python client.py
host is localhost, port is 1080
connected to  127.0.0.1
>> telnet send hello
received from server:  message received
>> telnet send hello2
received from server:  message received
>> telnet send hello3
received from server:  message received
>> telnet exec echo hello 4
received from server:  hello 4

>> telnet exec cd
received from server:  E:\uni\semester 6\Network\Projects\project 2\CN_Proj2_9731085

>> telnet history
received from server:
Command history:
telnet send hello
telnet send hello2
telnet send hello3
telnet exec echo hello 4
telnet exec cd
telnet history

>> telnet upload test.txt
file size:  14
received from server:  file received
>> telnet history
received from server:
Command history:
telnet send hello
telnet send hello2
telnet send hello3
telnet exec echo hello 4
telnet exec cd
telnet history
telnet upload test.txt
telnet history

>> telnet exit
received from server:  closing the connection...
PS E:\uni\semester 6\Network\Projects\project 2\CN_Proj2_9731085>
```

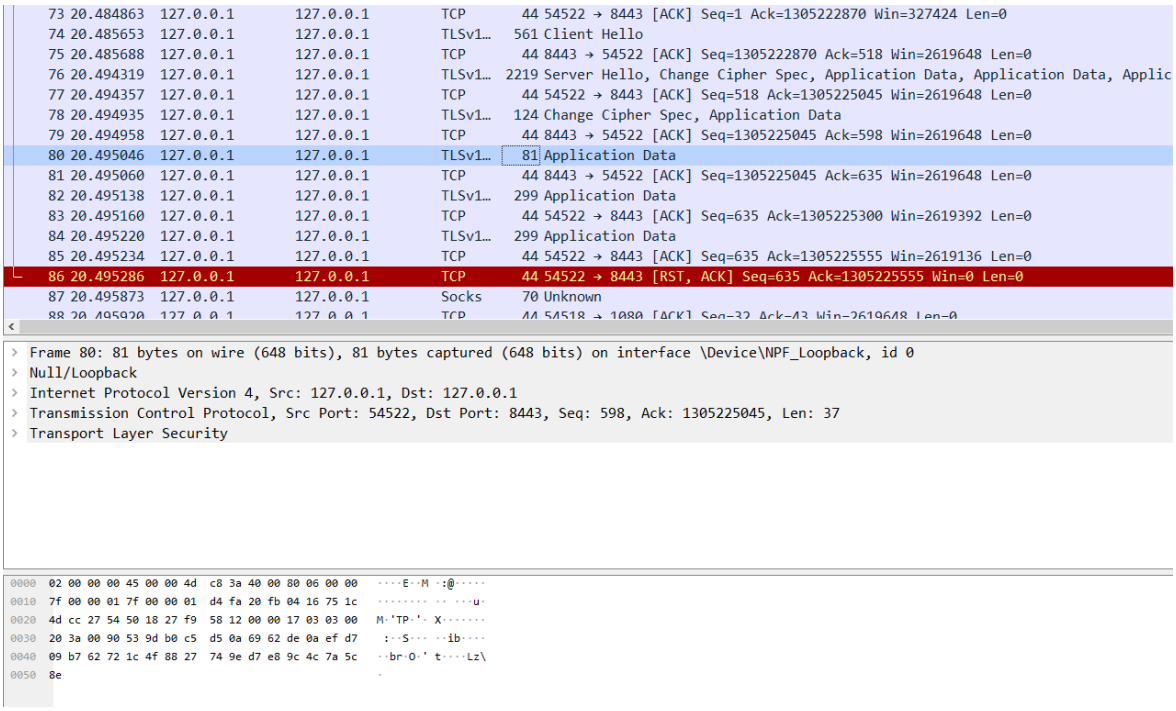
```
PS E:\uni\semester 6\Network\Projects\project 2\CN_Proj2_9731085> python server.py
client connected!
from client: telnet send hello
message is: hello
response is: message received
from client: telnet send hello2
message is: hello2
response is: message received
from client: telnet send hello3
message is: hello3
response is: message received
from client: telnet exec echo hello 4
command is: echo hello 4
response is: b'hello 4\r\n'
from client: telnet exec cd
command is: cd
response is: b'E:\\uni\\semester 6\\Network\\Projects\\project 2\\CN_Proj2_9731085\r\n'
from client: telnet history
from client: telnet upload test.txt
uploading...
file size:  14
response is: file received
from client: telnet history
from client: telnet exit
```

تحلیل بسته‌ها با وایرشارک:



همانطور که در تصویر مشخص است محتوای پیامی که رمزنگاری نشده‌است قابل مشاهده است.

اما پیامی که با TLS رمزنگاری شده‌است قابل مشاهده نمی‌باشد. (تصویر زیر)



73	20.484863	127.0.0.1	127.0.0.1	TCP	44	54522 → 8443 [ACK] Seq=1 Ack=1305222870 Win=327424 Len=0
74	20.485653	127.0.0.1	127.0.0.1	TLSv1…	561	Client Hello
75	20.485688	127.0.0.1	127.0.0.1	TCP	44	8443 → 54522 [ACK] Seq=1305222870 Ack=518 Win=2619648 Len=0
76	20.494319	127.0.0.1	127.0.0.1	TLSv1…	2219	Server Hello, Change Cipher Spec, Application Data, Application Data, Applic
77	20.494357	127.0.0.1	127.0.0.1	TCP	44	54522 → 8443 [ACK] Seq=518 Ack=1305225045 Win=2619648 Len=0
78	20.494935	127.0.0.1	127.0.0.1	TLSv1…	124	Change Cipher Spec, Application Data
79	20.494958	127.0.0.1	127.0.0.1	TCP	44	8443 → 54522 [ACK] Seq=1305225045 Ack=598 Win=2619648 Len=0
80	20.495046	127.0.0.1	127.0.0.1	TLSv1…	81	Application Data
81	20.495060	127.0.0.1	127.0.0.1	TCP	44	8443 → 54522 [ACK] Seq=1305225045 Ack=635 Win=2619648 Len=0
82	20.495138	127.0.0.1	127.0.0.1	TLSv1…	299	Application Data
83	20.495160	127.0.0.1	127.0.0.1	TCP	44	54522 → 8443 [ACK] Seq=635 Ack=1305225300 Win=2619392 Len=0
84	20.495220	127.0.0.1	127.0.0.1	TLSv1…	299	Application Data
85	20.495234	127.0.0.1	127.0.0.1	TCP	44	54522 → 8443 [ACK] Seq=635 Ack=1305225555 Win=2619136 Len=0
86	20.495286	127.0.0.1	127.0.0.1	TCP	44	54522 → 8443 [RST, ACK] Seq=635 Ack=1305225555 Win=0 Len=0
87	20.495873	127.0.0.1	127.0.0.1	Socks	70	Unknown
88	20.495920	127.0.0.1	127.0.0.1	TCP	44	54518 → 1080 [ACK] Seq=32 Ack=43 Win=2619648 Len=0

```
> Frame 80: 81 bytes on wire (648 bits), 81 bytes captured (648 bits) on interface \Device\NPF_Loopback, id 0
> Null/Loopback
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
> Transmission Control Protocol, Src Port: 54522, Dst Port: 8443, Seq: 598, Ack: 1305225045, Len: 37
> Transport Layer Security
```

```
0000  02 00 00 00 45 00 00 4d  c8 3a 40 00 80 06 00 00   ····E··M ·:@·····
0010  7f 00 00 01 7f 00 00 01  d4 fa 20 fb 04 16 75 1c   ········ ·· ···u·
0020  4d cc 27 54 50 18 27 f9  58 12 00 00 17 03 03 00   M·'TP·'· X·······
0030  20 3a 00 90 53 9d b0 c5  d5 0a 69 62 de 0a ef d7    :··S··· ··ib····
0040  09 b7 62 72 1c 4f 88 27  74 9e d7 e8 9c 4c 7a 5c   ··br·O·' t····Lz\
0050  8e                                                ·
```