مصطفی دریس پور پروژه شبکه
بخش کد:
پروژه شبکه سؤال اول:

به دلیل اینکه فایروال موجود در کلاینت این اجازه را به سرور نمی‌دهد تا پورتی بر روی کلاینت باز کند.
پس کلاینت نمی‌تواند پورتی بفرستد تا سرور به آن وصل شود فایروال جلوی آن را میگیرید.
بحث اضافه دیگر این است که شرکت های isp از یک ای پی ثابت برای چندین کاربر استفاده میکنند
به همین دلیل سرور نمی‌تواند پورتی را بر روی کلاینت بازکند.

سؤال دوم:
به این حمله path traversal *attack* می گویند.

بخش wireshark:

سؤال اول:

7	2.821759…	127.0.0.1	127.0.0.1	TCP	66	12345 → 46134 [ACK] Seq=293 Ack=5 Win=65536 L…
8	2.821976…	127.0.0.1	127.0.0.1	TCP	358	12345 → 46134 [PSH, ACK] Seq=293 Ack=5 Win=65…
9	2.822012…	127.0.0.1	127.0.0.1	TCP	66	46134 → 12345 [ACK] Seq=5 Ack=585 Win=65280 L…

در تصویر بالا Three Way Handshake کپچر شده است.

سؤال دوم:
الف)
tcp محدودیت ارسال 64kbyte دارد. و بالای آن را به قسمت‌های کوچکتر تقسیم می‌کند و می فرستد.
ب)

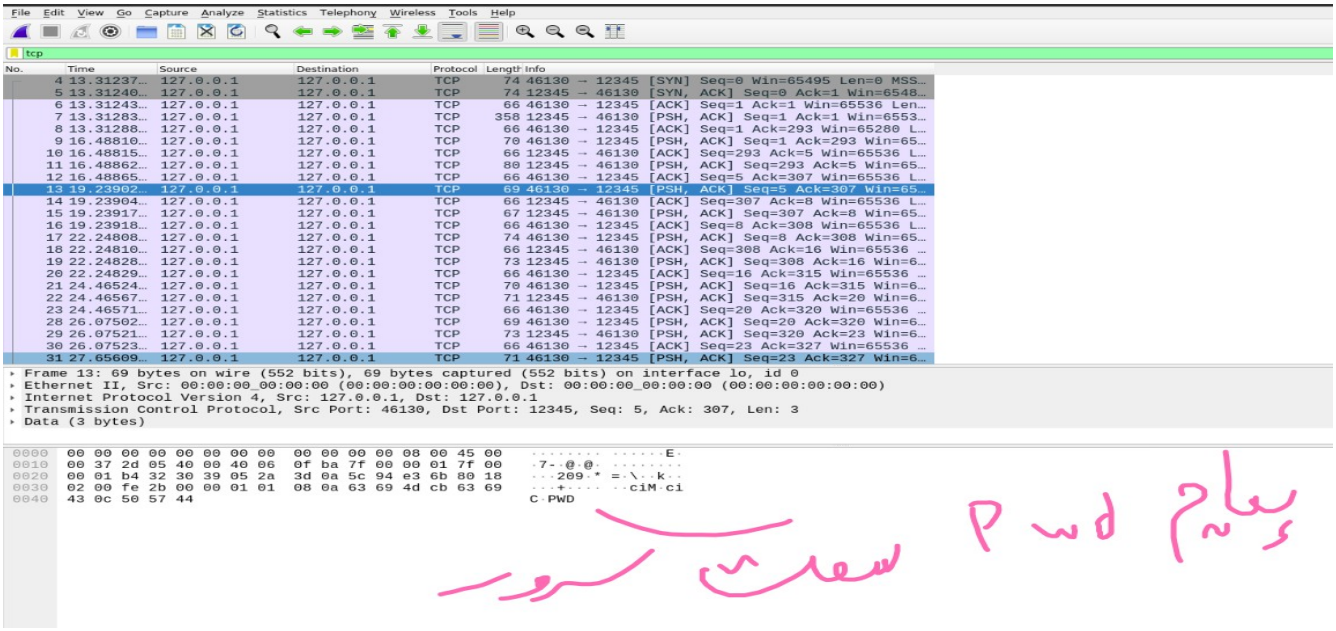
6	0.001247…	127.0.0.1	127.0.0.1	TCP	74	58768 → 4984 [SYN] Seq=0 Win=65495 Len=0 MSS=…
7	0.001301…	127.0.0.1	127.0.0.1	TCP	74	4984 → 58768 [SYN, ACK] Seq=0 Ack=1 Win=65483…
8	0.001351…	127.0.0.1	127.0.0.1	TCP	66	58768 → 4984 [ACK] Seq=1 Ack=1 Win=65536 Len=…
9	0.001904…	127.0.0.1	127.0.0.1	TCP	20…	4984 → 58768 [PSH, ACK] Seq=1 Ack=1 Win=65536…
10	0.001945…	127.0.0.1	127.0.0.1	TCP	66	58768 → 4984 [ACK] Seq=1 Ack=2025 Win=64000 L…
11	0.002008…	127.0.0.1	127.0.0.1	TCP	20…	4984 → 58768 [PSH, ACK] Seq=2025 Ack=1 Win=65…
12	0.002022…	127.0.0.1	127.0.0.1	TCP	66	58768 → 4984 [ACK] Seq=1 Ack=4049 Win=62976 L…
13	0.002070…	127.0.0.1	127.0.0.1	TCP	20…	4984 → 58768 [PSH, ACK] Seq=4049 Ack=1 Win=65…
14	0.002083…	127.0.0.1	127.0.0.1	TCP	66	58768 → 4984 [ACK] Seq=1 Ack=6073 Win=61952 L…
15	0.002122…	127.0.0.1	127.0.0.1	TCP	20…	4984 → 58768 [PSH, ACK] Seq=6073 Ack=1 Win=65…
16	0.002138…	127.0.0.1	127.0.0.1	TCP	66	58768 → 4984 [ACK] Seq=1 Ack=8097 Win=60928 L…
17	0.002193…	127.0.0.1	127.0.0.1	TCP	20…	4984 → 58768 [PSH, ACK] Seq=8097 Ack=1 Win=65…
18	0.002210…	127.0.0.1	127.0.0.1	TCP	66	58768 → 4984 [ACK] Seq=1 Ack=10121 Win=59904 …
19	0.002247…	127.0.0.1	127.0.0.1	TCP	20…	4984 → 58768 [PSH, ACK] Seq=10121 Ack=1 Win=6…
20	0.002264…	127.0.0.1	127.0.0.1	TCP	66	58768 → 4984 [ACK] Seq=1 Ack=12145 Win=58880 …
21	0.002322…	127.0.0.1	127.0.0.1	TCP	20…	4984 → 58768 [PSH, ACK] Seq=12145 Ack=1 Win=6…
22	0.002341…	127.0.0.1	127.0.0.1	TCP	66	58768 → 4984 [ACK] Seq=1 Ack=14169 Win=57856 …
23	0.002379…	127.0.0.1	127.0.0.1	TCP	20…	4984 → 58768 [PSH, ACK] Seq=14169 Ack=1 Win=6…
24	0.002394…	127.0.0.1	127.0.0.1	TCP	66	58768 → 4984 [ACK] Seq=1 Ack=16193 Win=56832 …
25	0.002446…	127.0.0.1	127.0.0.1	TCP	20…	4984 → 58768 [PSH, ACK] Seq=16193 Ack=1 Win=6…
26	0.002462…	127.0.0.1	127.0.0.1	TCP	66	58768 → 4984 [ACK] Seq=1 Ack=18217 Win=55808 …
27	0.002500…	127.0.0.1	127.0.0.1	TCP	20…	4984 → 58768 [PSH, ACK] Seq=18217 Ack=1 Win=6…
28	0.002516…	127.0.0.1	127.0.0.1	TCP	66	58768 → 4984 [ACK] Seq=1 Ack=20241 Win=54784 …
29	0.002571…	127.0.0.1	127.0.0.1	TCP	20…	4984 → 58768 [PSH, ACK] Seq=20241 Ack=1 Win=6…
30	0.002590…	127.0.0.1	127.0.0.1	TCP	66	58768 → 4984 [ACK] Seq=1 Ack=22265 Win=53888 …
31	0.002632…	127.0.0.1	127.0.0.1	TCP	20…	4984 → 58768 [PSH, ACK] Seq=22265 Ack=1 Win=6…
32	0.002649…	127.0.0.1	127.0.0.1	TCP	66	58768 → 4984 [ACK] Seq=1 Ack=24289 Win=52864 …
33	0.002704…	127.0.0.1	127.0.0.1	TCP	20…	4984 → 58768 [PSH, ACK] Seq=24289 Ack=1 Win=6…
34	0.002723…	127.0.0.1	127.0.0.1	TCP	66	58768 → 4984 [ACK] Seq=1 Ack=26313 Win=51840 …
35	0.002765…	127.0.0.1	127.0.0.1	TCP	20…	4984 → 58768 [PSH, ACK] Seq=26313 Ack=1 Win=6…
36	0.002781…	127.0.0.1	127.0.0.1	TCP	66	58768 → 4984 [ACK] Seq=1 Ack=28337 Win=50816 …
37	0.002834…	127.0.0.1	127.0.0.1	TCP	20…	4984 → 58768 [PSH, ACK] Seq=28337 Ack=1 Win=6…
38	0.002852…	127.0.0.1	127.0.0.1	TCP	66	58768 → 4984 [ACK] Seq=1 Ack=30361 Win=49792 …
39	0.002893…	127.0.0.1	127.0.0.1	TCP	20…	4984 → 58768 [PSH, ACK] Seq=30361 Ack=1 Win=6…
40	0.002908…	127.0.0.1	127.0.0.1	TCP	66	58768 → 4984 [ACK] Seq=1 Ack=32385 Win=48768 …
41	0.002985…	127.0.0.1	127.0.0.1	TCP	20…	4984 → 58768 [PSH, ACK] Seq=32385 Ack=1 Win=6…
42	0.003257…	127.0.0.1	127.0.0.1	TCP	32…	4984 → 58768 [ACK] Seq=34409 Ack=1 Win=65536 …
43	0.003361…	127.0.0.1	127.0.0.1	TCP	66	58768 → 4984 [ACK] Seq=1 Ack=67177 Win=48512 …
44	0.003414…	127.0.0.1	127.0.0.1	TCP	77…	4984 → 58768 [PSH, ACK] Seq=67177 Ack=1 Win=6…
45	0.003607…	127.0.0.1	127.0.0.1	TCP	32…	4984 → 58768 [ACK] Seq=74889 Ack=1 Win=65536 …
46	0.003678…	127.0.0.1	127.0.0.1	TCP	66	58768 → 4984 [ACK] Seq=1 Ack=107657 Win=8064 …

شروع فرایند دانلود از پورت رندم به صورت بالا شروع می شود
18 بسته ارسال شده است.

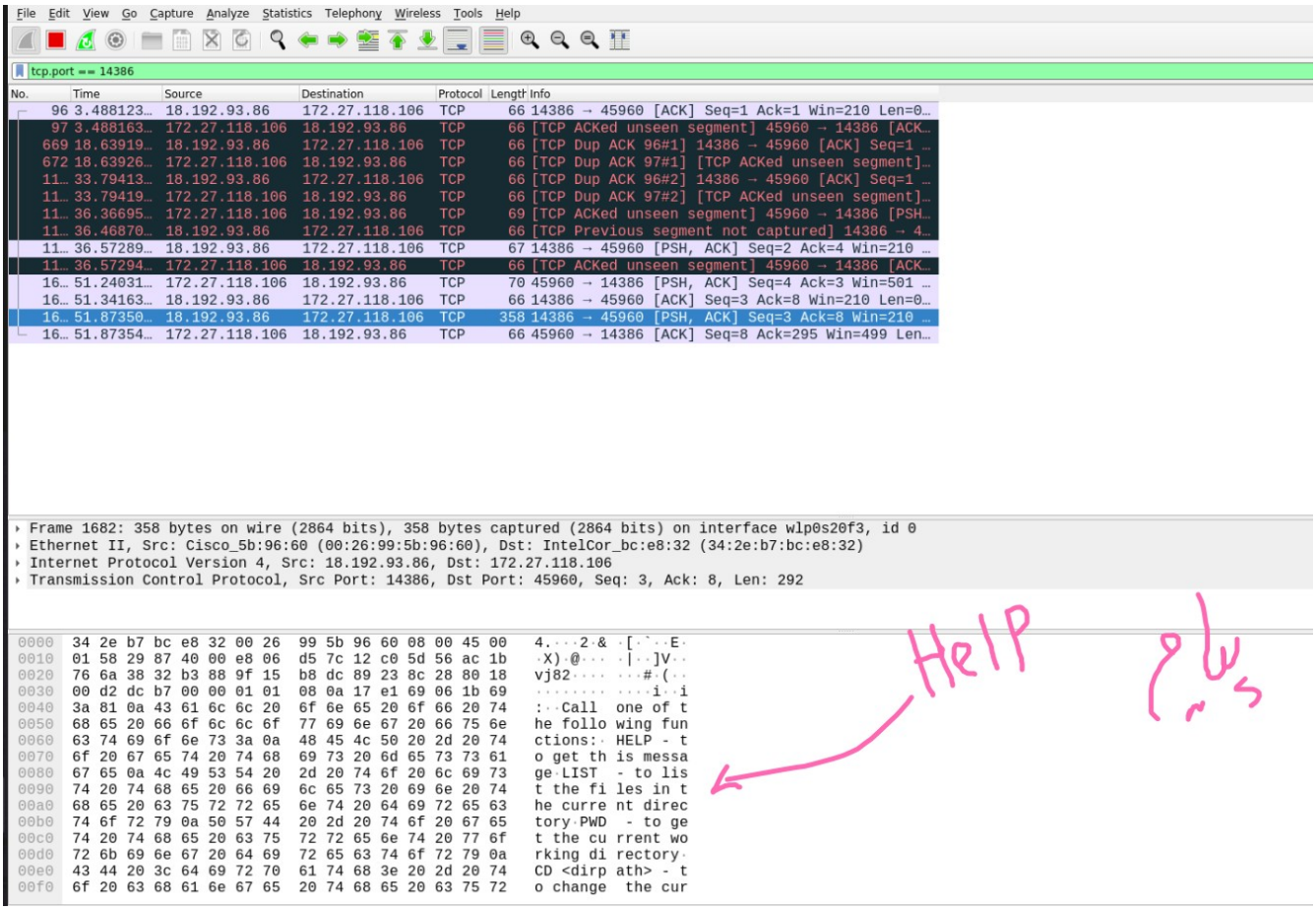
```
120 0.024510… 127.0.0.1   127.0.0.1   TCP   30… 4984 → 58768 [FIN, PSH, ACK] Seq=1254537 Ack=…
121 0.026293… 127.0.0.1   127.0.0.1   TCP    66 58768 → 4984 [FIN, ACK] Seq=1 Ack=1285091 Win…
122 0.026312… 127.0.0.1   127.0.0.1   TCP    66 4984 → 58768 [ACK] Seq=1285091 Ack=2 Win=6553…
```

و به این صورت ارتباط آن‌ها قطع می شود.(چون none-persistant است.)

بخش ngrok:



در تصویر بالا در wireshark قسمت loopback در سرور این پکت ها را دریافت می‌کنیم و پکت ها نشان داده برای مثال در تصویر بالا پیام دستور PWD از سمت کلاینت به سرور فرستاده شده است.



در تصویر بالا در wireshark بخش wifi می‌باشد که برای بررسی پکت های است که از سرور به کلاینت فرستاده می‌شود برای مثال در پیام بالا پیام help فرستاده شده است.
تفاوتی که با حالت قبل دارد این است که در حالت قبل تمام پیام‌ها در بخش loopback رد و بدل می‌شود اما با استفاده از ngrok این پیام‌ها برای کلاینت در بخش wifi (بستگی به نوع اتصال) و در اینترنت رد و بدل می‌شد که همانطورر که در تصویر بالا نشان داده شده است.
برای بخش سرور هم کماکان پیام‌ها در بخش loopback ارسال می شد.

در ابتدا بدون هیچ تغییری این امر ممکن نیست زیرا نیاز است پورت جدیدی باز شود و ngrok تنها یک پورت در اختیار ما می گذارد.

شیوه پیاده‌سازی ngrok برای download کردن:
ابتدا در سرور یک دستور ngrok tcp random_port (با کتابخانه sys) میزنیم و هاست و پورت را برای کلاینت می فرستیم
و سپس کلاینت به آن port و host که فرستاده شده وصل می‌شود و فایل را دریافت می‌کند سپس آن اتصال بسته می شود.

نکته: اجرای دستور ngrok tcp random_port در سرور انجام می شود. و خروجی آنکه نام هاست و پورت است برای کلاینت فرستاده می‌شود تا به آن متصل شود.