# חלק א:

קובץ 1ex2.pcapng:

תחילה הרצנו את הקוד של השרת והסרבר ממצגת התרגול והסנפנו בwireshark:

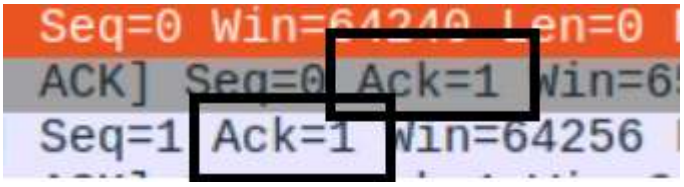
| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 1 | 0.000000000 | 10.0.2.5 | 10.0.2.4 | TCP | 74 | 56748 → 12345 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM T… |
| 2 | 0.000025649 | 10.0.2.4 | 10.0.2.5 | TCP | 74 | 12345 → 56748 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 … |
| 3 | 0.002915104 | 10.0.2.5 | 10.0.2.4 | TCP | 66 | 56748 → 12345 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=11318297… |
| 4 | 0.037961741 | 10.0.2.5 | 10.0.2.4 | TCP | 77 | 56748 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=11 TSval=11… |
| 5 | 0.038007073 | 10.0.2.4 | 10.0.2.5 | TCP | 66 | 12345 → 56748 [ACK] Seq=1 Ack=12 Win=65152 Len=0 TSval=2838028… |
| 6 | 0.038243884 | 10.0.2.4 | 10.0.2.5 | TCP | 77 | 12345 → 56748 [PSH, ACK] Seq=1 Ack=12 Win=65152 Len=11 TSval=2… |
| 7 | 0.039013209 | 10.0.2.5 | 10.0.2.4 | TCP | 66 | 56748 → 12345 [ACK] Seq=12 Ack=12 Win=64256 Len=0 TSval=113182… |
| 8 | 0.040797375 | 10.0.2.5 | 10.0.2.4 | TCP | 86 | 56748 → 12345 [PSH, ACK] Seq=12 Ack=12 Win=64256 Len=20 TSval=… |
| 9 | 0.042851421 | 10.0.2.4 | 10.0.2.5 | TCP | 66 | 12345 → 56748 [FIN, ACK] Seq=12 Ack=32 Win=65152 Len=0 TSval=2… |
| 10 | 0.058422000 | 10.0.2.5 | 10.0.2.4 | TCP | 66 | 56748 → 12345 [FIN, ACK] Seq=32 Ack=13 Win=64256 Len=0 TSval=1… |
| 11 | 0.058445750 | 10.0.2.4 | 10.0.2.5 | TCP | 66 | 12345 → 56748 [ACK] Seq=13 Ack=33 Win=65152 Len=0 TSval=283802… |

```
74 56748 → 12345 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM T…
74 12345 → 56748 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 …
66 56748 → 12345 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=11318297…
```

ניתן לראות את תחילת התקשורת עם לחיצת הידיים בה הלקוח מתחיל את התקשורת בעזרת SYN (רואים שליחה מפורט 56748 כיוון שהלקוח לא עשה bind ומערכת ההפעלה בחרה לו פורט שרירותית). הSYN זו בקשת התחברות לשרת.

השרת שולח ללקוח מפורט 12345 ACK לSYN ונוסף פאנטום ביט לack number מצד השרת.

לאחר מכן הלקוח שולח לשרת ACK על הsyn ack וגם לו נוסיף פאנטום ביט לack number.




ניתן לראות לאחר פתיחת הפאקטה הראשונה את הפרטים בתחילית ה-TCP.

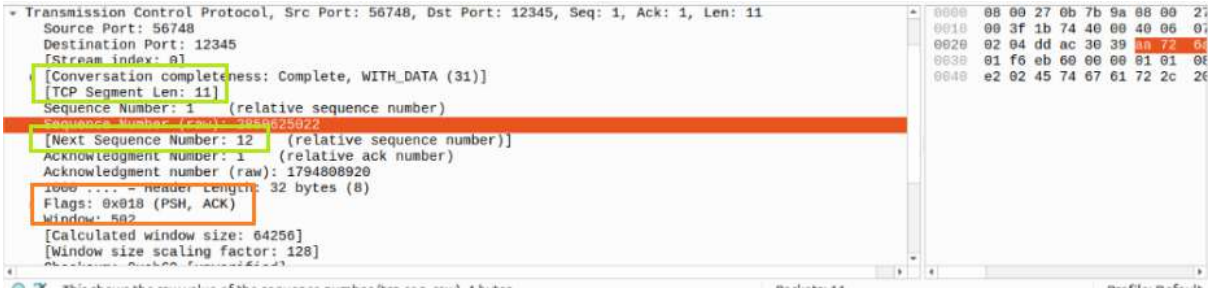
בריבוע הכתום ניתן לראות את ה-sequence number הרלטיבי והלא רלטיבי. בגלל שזוהי ההודעה הראשונה בתקשורת, המספר הרלטיבי הוא 0.

ניתן לראות בירוק את הדגל היחיד שדולקת SYN. הסיבה שדגל הACK לא דולק היא שזוהי הודעה ראשונה בתקשורת

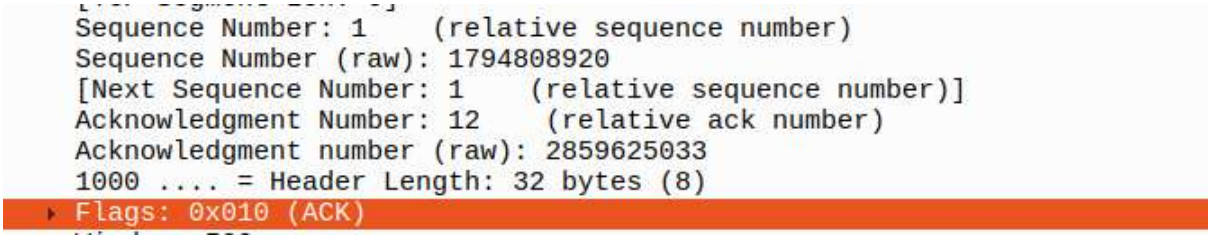
```
Sequence Number: 0    (relative sequence number)
Sequence Number (raw): 2063013436
[Next Sequence Number: 1    (relative sequence number)]
Acknowledgment Number: 0
Acknowledgment number (raw): 0
1010 .... = Header Length: 40 bytes (10)
Flags: 0x002 (SYN)
```

```
Flags: 0x002 (SYN)
    000. .... .... = Reserved: Not set
    ...0 .... .... = Accurate ECN: Not set
    .... 0... .... = Congestion Window Reduced: Not set
    .... .0.. .... = ECN-Echo: Not set
    .... ..0. .... = Urgent: Not set
    .... ...0 .... = Acknowledgment: Not set
    .... .... 0... = Push: Not set
    .... .... .0.. = Reset: Not set
    .... .... ..1. = Syn: Set
    .... .... ...0 = Fin: Not set
```

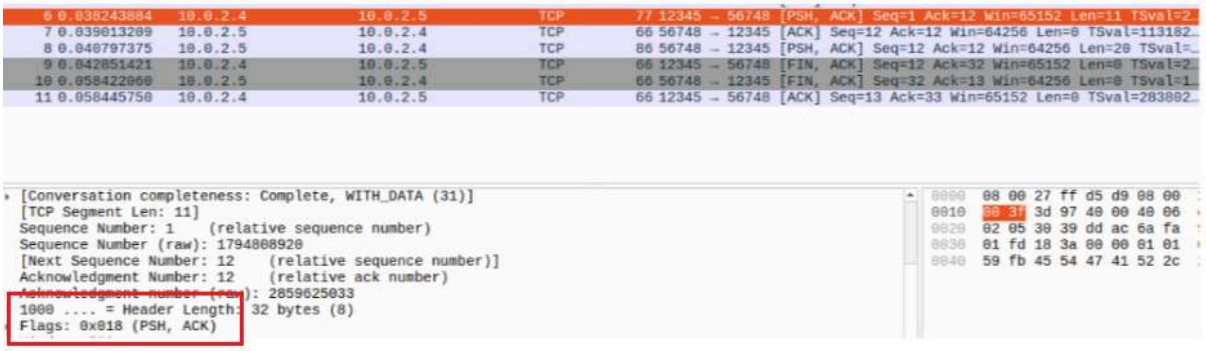
ניתן לראות בתחילית הפאקטה הרביעית את שליחת ההודעה של שמותינו מהלקוח אל השרת:



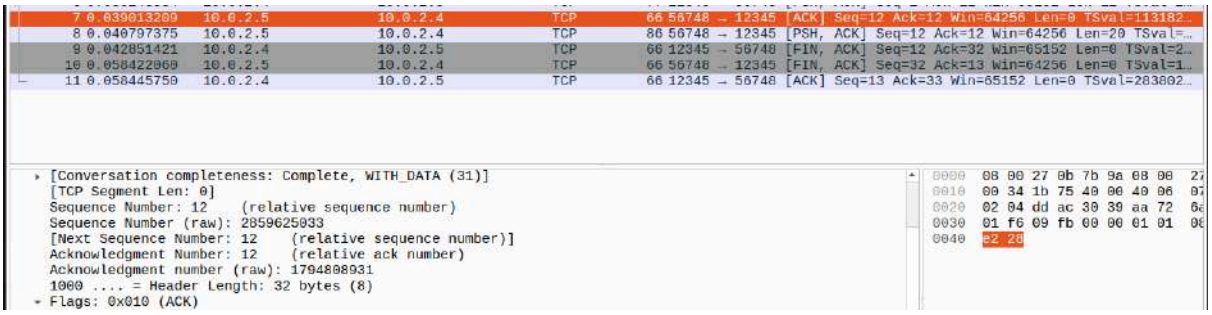

בריבועים הירוקים אפשר לראות ששלחנו מידע בגודל 11 בתים (Etgar, Ofry) וכן ה-wireshark מנחש שמספר ה-sequence הבא (לאחר שהשרת יאשר שהוא קיבל) יהיה 12 (רלטיבי) כלומר 3859625034.
11 בתים עבור מידע ששלח כעת ועוד הפאנטום ביט.
בריבוע הכתום ניתן לראות את הדגלים שדולקים בתחילית. ACK ו-PSH כי הלקוח שלח מידע חדש.
לאחר מכן, בפאקטה החמישית, השרת שולח הודעה עבור קבלת המידע. ניתן לראות
שה-acknowledgment number של השרת עלה ל-12 רלטיבית כי התקבלו אצלו 12 בתים.
המספר הגולמי אצל השרת הוא בעצם 2859625033.




כעת השרת שולח ללקוח את אותה הודעה באותיות גדולות. ניתן לראות את דגל ה-PSH דולק




כעת, ניתן לראות שקיבלנו ack וה-acknowledgment number עלה ל-12 יחסי ול-2859625033 כאשר לפני זה הוא היה 2859625022 שזה בדיוק 11 בתים (גודל ההודעה) פחות מלאחר קבלת ההודעה מהלקוח.



עכשיו ניתן לראות את ההודעה השניה שנשלחה מהלקוח לשרת (תעודות הזהות שלנו). ניתן לראות שהודעה זו היא באורך 20 בתים.

```
8 0.040797375   10.0.2.5   10.0.2.4   TCP   86 56748 → 12345 [PSH, ACK] Seq=12 Ack=12 Win=64256 Len=20 TSval=…
9 0.042851421   10.0.2.4   10.0.2.5   TCP   66 12345 → 56748 [FIN, ACK] Seq=12 Ack=32 Win=65152 Len=0 TSval=2…
10 0.058422060  10.0.2.5   10.0.2.4   TCP   66 56748 → 12345 [FIN, ACK] Seq=32 Ack=13 Win=64256 Len=0 TSval=1…
11 0.058445750  10.0.2.4   10.0.2.5   TCP   66 12345 → 56748 [ACK] Seq=13 Ack=33 Win=65152 Len=0 TSval=283802…

Source Port: 56748
Destination Port: 12345
[Stream index: 0]
▸ [Conversation completeness: Complete, WITH_DATA (31)]
[TCP Segment Len: 20]
Sequence Number: 12    (relative sequence number)
Sequence Number (raw): 2859625033
[Next Sequence Number: 32    (relative sequence number)]
Acknowledgment Number: 12    (relative ack number)
Acknowledgment number (raw): 1794808931
1000 .... = Header Length: 32 bytes (8)
▾ Flags: 0x018 (PSH, ACK)
  000. .... .... = Reserved: Not set

0000  08 00 45 00  ··'{··'····E·
0010  02 05 0a 00  ·H·v@·@··2····
0020  9c 63 80 18  ···69·r jIj··c··
0030  59 fd a9 28  ·····CvY··(
0040  20 32 30 39  (209520 931, 209
                   130251
```

לאחר מכן השרת מסיים את החיבור ומאותת שקיבל את ההודעה עם תעודות הזהות באותה הפאקטה (ניתן לראות גם את דגל ה-ACK וגם את דגל ה-FIN דולקים)

```
9 0.042851421   10.0.2.4   10.0.2.5   TCP   66 12345 → 56748 [FIN, ACK] Seq=12 Ack=32 Win=65152 Len=0 TSval=2…
10 0.058422060  10.0.2.5   10.0.2.4   TCP   66 56748 → 12345 [FIN, ACK] Seq=32 Ack=13 Win=64256 Len=0 TSval=1…
11 0.058445750  10.0.2.4   10.0.2.5   TCP   66 12345 → 56748 [ACK] Seq=13 Ack=33 Win=65152 Len=0 TSval=283802…

Source Port: 12345
Destination Port: 56748
[Stream index: 0]
▸ [Conversation completeness: Complete, WITH_DATA (31)]
[TCP Segment Len: 0]
Sequence Number: 12    (relative sequence number)
Sequence Number (raw): 1794808931
[Next Sequence Number: 13    (relative sequence number)]
Acknowledgment Number: 32    (relative ack number)
Acknowledgment number (raw): 2859625053
1000 .... = Header Length: 32 bytes (8)
▾ Flags: 0x011 (FIN, ACK)
  000. .... .... = Reserved: Not set
  ...0 .... .... = Accurate ECN: Not set
  .... 0... .... = Congestion Window Reduced: Not set
  .... .0.. .... = ECN-Echo: Not set

0000  08 00 27 ff d5 d9 08 00
0010  00 34 3d 98 40 00 40 06
0020  02 05 30 39 dd ac 6a fa
0030  01 fd 18 2f 00 00 01 01
0040  59 fd
```

```
▾ Flags: 0x011 (FIN, ACK)
  000. .... .... = Reserved: Not set
  ...0 .... .... = Accurate ECN: Not set
  .... 0... .... = Congestion Window Reduced: Not set
  .... .0.. .... = ECN-Echo: Not set
  .... ..0. .... = Urgent: Not set
  .... ...1 .... = Acknowledgment: Set
  .... .... 0... = Push: Not set
  .... .... .0.. = Reset: Not set
  .... .... ..0. = Syn: Not set
▸ .... .... ...1 = Fin: Set
```

הלקוח מסיים גם כן את החיבור עם הודעת FIN ו-ACK. ניתן לראות גם כאן פאנטום ביט

```
10 0.058422060  10.0.2.5   10.0.2.4   TCP   66 56748 → 12345 [FIN, ACK] Seq=32 Ack=13 Win=64256 L
11 0.058445750  10.0.2.4   10.0.2.5   TCP   66 12345 → 56748 [ACK] Seq=13 Ack=33 Win=65152 Len=0

▾ Transmission Control Protocol, Src Port: 56748, Dst Port: 12345, Seq: 32, Ack: 13, Len: 0
  Source Port: 56748
  Destination Port: 12345
  [Stream index: 0]
  ▸ [Conversation completeness: Complete, WITH_DATA (31)]
  [TCP Segment Len: 0]
  Sequence Number: 32    (relative sequence number)
  Sequence Number (raw): 2859625053
  [Next Sequence Number: 33    (relative sequence number)]
  Acknowledgment Number: 13    (relative ack number)
  Acknowledgment number (raw): 1794808932
  1000 .... = Header Length: 32 bytes (8)
  ▾ Flags: 0x011 (FIN, ACK)
    000. .... .... = Reserved: Not set
    ...0 .... .... = Accurate ECN: Not set
    .... 0... .... = Congestion Window Reduced: Not set
    .... .0.. .... = ECN-Echo: Not set
    .... ..0. .... = Urgent: Not set
    .... ...1 .... = Acknowledgment: Set
    .... .... 0... = Push: Not set
    .... .... .0.. = Reset: Not set
    .... .... ..0. = Syn: Not set
  ▸ .... .... ...1 = Fin: Set
  ▸ [TCP Flags: ·····A···F]

0000  08 00 27 0b
0010  00 34 1b 77
0020  02 04 dd ac
0030  01 f6 09 d5
0040  e2 2d
```

```
Seq=12 Ack=12
Seq=12 Ack=32
Seq=32 Ack=13
```

פאנטום ביט - עלינו מ12 ל13.

```
11 0.058445750   10.0.2.4   10.0.2.5   TCP   66 12345 → 56748 [ACK] Seq=13 Ack=33 Win=65152 Len=0 TSval=283

Transmission Control Protocol, Src Port: 12345, Dst Port: 56748, Seq: 13, Ack: 33, Len: 0
    Source Port: 12345
    Destination Port: 56748
    [Stream index: 0]
  ▸ [Conversation completeness: Complete, WITH_DATA (31)]
    [TCP Segment Len: 0]
    Sequence Number: 13    (relative sequence number)
    Sequence Number (raw): 1794808932
    [Next Sequence Number: 13    (relative sequence number)]
    Acknowledgment Number: 33    (relative ack number)
    Acknowledgment number (raw): 2859625054
    1000 .... = Header Length: 32 bytes (8)
  ▾ Flags: 0x010 (ACK)
      000. .... .... = Reserved: Not set
      ...0 .... .... = Accurate ECN: Not set
      .... 0... .... = Congestion Window Reduced: Not set
      .... .0.. .... = ECN-Echo: Not set
      .... ..0. .... = Urgent: Not set
      .... ...1 .... = Acknowledgment: Set
      .... .... 0... = Push: Not set
      .... .... .0.. = Reset: Not set
      .... .... ..0. = Syn: Not set
      .... .... ...0 = Fin: Not set

0000  08 00 27 ff d5 d9 08 00
0010  00 34 3d 99 40 00 40 06
0020  02 05 30 39 dd ac 6a fa
0030  01 fd 18 2f 00 00 01 01
0040  5a 07
```

כעת השרת שולח ACK על הודעת הFIN מהלקוח. ניתן לראות גם כאן פאנטום ביט מ32 ל33 ב-ACK.

```
N, ACK] Seq=32 Ac
K] Seq=13 Ack=33
```

או בעצם עלה מ-2859625053 ל-2859625054.

הקוד של השרת:

```python
import socket
server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server.bind(('', 12345))
server.listen(5)

while True:
    client_socket, client_address = server.accept()
    print('Connection from: ', client_address)
    data = client_socket.recv(100)
    print('Received: ', data)
    client_socket.send(data.upper())
    data = client_socket.recv(100)
    print('Received: ', data)
    client_socket.close()
    print('Client disconnected')
```

הקוד של הלקוח:

```python
import socket
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect(('10.0.2.4', 12345))
s.send(b'Etgar, Ofry')
data = s.recv(100)
print("Server sent: ", data)
s.send(b'209520931, 209130251')
s.close()
```

# הסבר גרסאות הקוד השונות:

**:V1**

בגרסה זו הלקוח פותח שני sockets כדי לתקשר עם השרת שהמידע שלו הועבר דרך ארגומנטים.
בסוקט השני הוא שולח הודעה אחת, לאחר מכן ישן 5 שניות ואז שולח הודעה אחרת לסוקט הראשון שהוא פתח.
הוא מצפה לקבל מהשרת תשובה חזרה קודם כל על המחרוזת השניה שהוא שלח ואז סוגר את הסוקט. לאחר מכן מצפה לקבל תשובה על ההודעה הראשונה ששלח ולאחר מכן גם כן סוגר את הסוקט.

השרת פותח סוקט TCP ומאזין לכל IP ולפורט שניתן כארגומנט. הוא מאפשר האזנה ל0 משתמשים שממתינים. כלומר השרת תפוס, הוא לא יאפשר שלקוח אחר יחכה.
השרת מנסה להתחבר ללקוח ולאחר מכן מדפיס את פרטיו. לאחר מכן הוא מצפה לקבל עוד חיבור וגם כן מדפיס את פרטיו.
השרת מקבל הודעה מהחיבור השני, מדפיס אותו ואז שולח חלק ממנו (אינקס 0 עד 7) בחזרה לחיבור הראשון.
לאחר מכן, השרת מקבל הודעה מהלקוח הראשון, מדפיס אותה ואז שולח לחיבור השני חלק מההודעה (אינדקס 0 עד 5)
לאחר מכן השרת סוגר את שני החיבורים.

ניתן לראות בריבוע הורוד כי החיבור הראשון הצליח - יש לחיצת ידיים.
בריבוע התכלת ניתן לראות שכנראה כאשר הלקוח לשרת בקשת syn

ניתן לראות את לחיצת הידיים ב2 החיבורים הראשונים.
זה קורה כי הלקוח פותח 2 חיבורים עם השרת

| No. | Time | Source | Destination | Proto | Leng | Info |
|---|---|---|---|---|---|---|
| 24 | 5.687526671 | 10.0.2.4 | 10.0.2.5 | TCP | 74 | 42456 → 12345 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=2273174999 TSecr=0 WS=128 |
| 25 | 5.687560604 | 10.0.2.5 | 10.0.2.4 | TCP | 74 | 12345 → 42456 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=2410781733 TSecr=… |
| 26 | 5.688477140 | 10.0.2.4 | 10.0.2.5 | TCP | 66 | 42456 → 12345 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=2273175000 TSecr=2410781733 |
| 27 | 5.690273228 | 10.0.2.4 | 10.0.2.5 | TCP | 74 | 42458 → 12345 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=2273175002 TSecr=0 WS=128 |
| 28 | 5.690296010 | 10.0.2.5 | 10.0.2.4 | TCP | 74 | 12345 → 42458 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=2410781733 TSecr=… |
| 29 | 5.691271496 | 10.0.2.4 | 10.0.2.5 | TCP | 66 | 42458 → 12345 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=2273175003 TSecr=2410781733 |

לאחר מכן הלקוח שולח באחד החיבורים הודעה לשרת ומקבל ack (צהוב)
השרת שולח דרך החיבור השני את ההודעה ללקוח (ירוק) ומקבל ממנו ack
לאחר מכן, מכיוון שהלקוח ישן 5 שניות לפני שליחת הודעה בחיבור השני, רואים כי בזמן 10 שניות החיבור השני שולח לשרת הודעה, ומקבל ack (תכלת)
השרת שולח לחיבור הראשון הודעה (ורוד) ומקבל ack (לאחר ששלח סגירת הודעה לחיבור השני)

| | | | | | | |
|---|---|---|---|---|---|---|
| 30 | 5.692083779 | 10.0.2.4 | 10.0.2.5 | TCP | 77 | 42458 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=642 |
| 31 | 5.692097584 | 10.0.2.5 | 10.0.2.4 | TCP | 66 | 12345 → 42458 [ACK] Seq=1 Ack=12 Win=64256 L |
| 32 | 5.692680136 | 10.0.2.5 | 10.0.2.4 | TCP | 73 | 12345 → 42456 [PSH, ACK] Seq=1 Ack=1 Win=642 |
| 33 | 5.693143091 | 10.0.2.4 | 10.0.2.5 | TCP | 66 | 42456 → 12345 [ACK] Seq=1 Ack=8 Win=64256 Le |
| 63 | 10.709038403 | 10.0.2.4 | 10.0.2.5 | TCP | 76 | 42456 → 12345 [PSH, ACK] Seq=1 Ack=8 Win=642 |
| 64 | 10.709065432 | 10.0.2.5 | 10.0.2.4 | TCP | 66 | 12345 → 42456 [ACK] Seq=8 Ack=11 Win=64256 L |
| 65 | 10.710998398 | 10.0.2.5 | 10.0.2.4 | TCP | 71 | 12345 → 42458 [PSH, ACK] Seq=1 Ack=12 Win=64 |
| 66 | 10.712304110 | 10.0.2.5 | 10.0.2.4 | TCP | 66 | 12345 → 42456 [FIN, ACK] Seq=8 Ack=11 Win=64 |
| 67 | 10.718311729 | 10.0.2.4 | 10.0.2.5 | TCP | 66 | 42458 → 12345 [ACK] Seq=12 Ack=6 Win=64256 L |

כפי שאמרנו, השרת סוגר את החיבור השני, החיבור השני גם כן סוגר את החיבור ועל הדרך שולח ACK על הסגירה של השרת.
השרת שולח ACK על הסגירה של הלקוח.
לאחר מכן הלקוח סוגר את החיבור הראשון והשרת שולח ack.

| | | | | | | |
|---|---|---|---|---|---|---|
| 66 | 10.712304110 | 10.0.2.5 | 10.0.2.4 | TCP | 66 | 12345 → 42456 [FIN, ACK] Seq=8 Ack=11 Win=64256 |
| 67 | 10.718311729 | 10.0.2.4 | 10.0.2.5 | TCP | 66 | 42458 → 12345 [ACK] Seq=12 Ack=6 Win=64256 Len=0 |
| 68 | 10.725413017 | 10.0.2.4 | 10.0.2.5 | TCP | 66 | 42456 → 12345 [FIN, ACK] Seq=11 Ack=9 Win=64256 |
| 69 | 10.725447134 | 10.0.2.5 | 10.0.2.4 | TCP | 66 | 12345 → 42456 [ACK] Seq=9 Ack=12 Win=64256 Len=0 |
| 70 | 10.736050841 | 10.0.2.4 | 10.0.2.5 | TCP | 66 | 42458 → 12345 [FIN, ACK] Seq=12 Ack=6 Win=64256 |
| 71 | 10.781219434 | 10.0.2.5 | 10.0.2.4 | TCP | 66 | 12345 → 42458 [ACK] Seq=6 Ack=13 Win=64256 Len=0 |

**:V2**

בגרסה זו הלקוח מקבל פרטי שרת כארגומנטים, פותח סוקט TCP ומתחבר לשרת הנתון. הוא שולח לשרת מחרוזת שהגדיר (`'!World! Hello, World'`) ומצפה לקבל בחזרה הודעה.
לאחר מכן הלקוח סוגר את החיבור ומדפיס את ההודעה שהתקבלה מהשרת.

השרת גם כן פותח חיבור TCP ומאזין לכל כתובת IP ופורט שהתקבל כארגומנט. מאפשר האזנה ללקוח 1 ממתין ואז רץ בלולאה.
הוא מקבל חיבור מדפיס את פרטי החיבור ואז שוב בלולאה מקבל את המידע הנשלח מהלקוח בטפטופים של 5 בתים. בכל טפטוף בזה הוא מדפיס את המידע שהתקבל (5 בתים), ושולח אותו חזרה באותיות גדולות.

לאחר שנגמר כל המידע שיש לקרוא, השרת סוגר את החיבור ועובר ללקוח הבא

ניתן לראות שהלקוח התחבר לשרת בלחיצת יד משולשת
לאחר מכן (בצהוב) הלקוח שולח לשרת הודעה שמתקבלת ע"י ACK. השרת גם כן שולח ללקוח תשובה (בירוק), אבל כפי שהסברנו, השרת קורא ושולח בפטפוטים.
הלקוח סוגר את החיבור לאחר קריאת המידע הראשון ששלח השרת (בחום) אך השרת לא הספיק לשלוח את כל המידע ולכן שולח (בורוד) עוד מידע ללקוח לאחר סגירת הסוקט מצד הלקוח, על כן מתקבל RST.

| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 77 | 7.227940974 | 10.0.2.4 | 10.0.2.5 | TCP | 74 | 43292 → 12345 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PER |
| 78 | 7.227964478 | 10.0.2.5 | 10.0.2.4 | TCP | 74 | 12345 → 43292 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=14 |
| 79 | 7.228884858 | 10.0.2.4 | 10.0.2.5 | TCP | 66 | 43292 → 12345 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=22678 |
| 80 | 7.230102164 | 10.0.2.4 | 10.0.2.5 | TCP | 86 | 43292 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=20 TSval |
| 81 | 7.230119054 | 10.0.2.5 | 10.0.2.4 | TCP | 66 | 12345 → 43292 [ACK] Seq=1 Ack=21 Win=65152 Len=0 TSval=2405 |
| 82 | 7.230682875 | 10.0.2.5 | 10.0.2.4 | TCP | 71 | 12345 → 43292 [PSH, ACK] Seq=1 Ack=21 Win=65152 Len=5 TSval |
| 83 | 7.231821884 | 10.0.2.4 | 10.0.2.5 | TCP | 66 | 43292 → 12345 [ACK] Seq=21 Ack=6 Win=64256 Len=0 TSval=2267 |
| 84 | 7.231822075 | 10.0.2.4 | 10.0.2.5 | TCP | 66 | 43292 → 12345 [FIN, ACK] Seq=21 Ack=6 Win=64256 Len=0 TSval |
| 85 | 7.231840004 | 10.0.2.5 | 10.0.2.4 | TCP | 81 | 12345 → 43292 [PSH, ACK] Seq=6 Ack=21 Win=65152 Len=15 TSva |
| 86 | 7.232485846 | 10.0.2.5 | 10.0.2.4 | TCP | 66 | 12345 → 43292 [FIN, ACK] Seq=21 Ack=22 Win=65152 Len=0 TSva |
| 87 | 7.232927072 | 10.0.2.4 | 10.0.2.5 | T | 60 | 43292 → 12345 [RST] Seq=21 Win=0 Len=0 |
| 88 | 7.234128442 | 10.0.2.4 | 10.0.2.5 | [illegible] | 60 | 43292 → 12345 [RST] Seq=22 Win=0 Len=0 |

**:V3**

קליינט - מקבלים בארגומנט TCP_IP וTCP_PORT, יוצרים סוקט TCP, יוצרים חיבור לפי הארגומנטים שהתקבלו. שולחים כהודעה 'hello world' ואז מקבלים כהודעה מקסימום 1024 בתים. ואז מקבלים שוב פעם מקסימום 1024 בתים. מדפיסים אחרי כל קבלת מידע. וסוגרים את החיבור.

שרת: יוצרים סוקט TCP, עושים בינד כלומר מגדירים שמאזינים לIP=0.0.0.0 כלומר מגדירים שאפשר לפנות אלינו מכל IP, וגם מגדירים בבינד את הפורט שלנו שקיבלנו אותו דרך הארגומנטים.
מגדירים שמקשיבים ל1 כלומר יוצרים תור המתנה שרק אחד קליינט אחד יכול לשבת בו ולהמתין.
בלולאה אינסופית מבצעים את הפעולות הבאות:
יוצרים חיבור עם קליינט, מדפיסים את הכתובת של הקליינט כלומר את הIP והPORT שלו.

ואז בלולאה אינסופית - מקבלים מידע מהקליינט, אם לא חוזר כלום (זה קורה כשהבאפר ריק) יוצאים מהלולאה ואז סוגרים את הconnection אם חוזר מידע שולחים את המידע חזרה עם אותיות גדולות 1000 פעמים.
(נשים לב שהשרת שולח ללקוח יותר מידע משהוא קורא ולכן נראה זאת בתעבורת הwireshark).

ניתן לראות את שליחת המידע (בצהוב) מהלקוח לשרת, (בירוק) מהשרת ללקוח. נשלחו 13000 בתים מהשרת ללקוח ולכן המידע התפצל לשני סגמנטים. לכן, (תכלת) התקבלו גם שני ACK-ים מהלקוח לשרת.
הלקוח קורא (בקוד עצמו) פעמיים 1024 בתים (שזה פחות מהכמות שהשרת שולח) ולכן בסוף נשלח RST מהלקוח לשרת כדי לתת לשרת חיווי ששכבת האפליקציה לא קראה את כל המידע שנשלח אליה.

tcp.port == 12345

| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 63 | 6.410223420 | 10.0.2.4 | 10.0.2.5 | TCP | 74 | 36610 → 12345 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM T… |
| 64 | 6.410251615 | 10.0.2.5 | 10.0.2.4 | TCP | 74 | 12345 → 36610 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 … |
| 65 | 6.410931284 | 10.0.2.4 | 10.0.2.5 | TCP | 66 | 36610 → 12345 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=22681266… |
| 66 | 6.413834727 | 10.0.2.4 | 10.0.2.5 | TCP | 79 | 36610 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=13 TSval=22… |
| 67 | 6.413886428 | 10.0.2.5 | 10.0.2.4 | TCP | 66 | 12345 → 36610 [ACK] Seq=1 Ack=14 Win=65152 Len=0 TSval=2405733… |
| 68 | 6.414547965 | 10.0.2.5 | 10.0.2.4 | TCP | 7306 | 12345 → 36610 [PSH, ACK] Seq=1 Ack=14 Win=65152 Len=7240 TSval… |
| 69 | 6.415244328 | 10.0.2.5 | 10.0.2.4 | TCP | 5826 | 12345 → 36610 [PSH, ACK] Seq=7241 Ack=14 Win=65152 Len=5760 TS… |
| 70 | 6.416218353 | 10.0.2.4 | 10.0.2.5 | TCP | 66 | 36610 → 12345 [ACK] Seq=14 Ack=7241 Win=78592 Len=0 TSval=2268… |
| 71 | 6.416218712 | 10.0.2.4 | 10.0.2.5 | TCP | 66 | 36610 → 12345 [ACK] Seq=14 Ack=13001 Win=78592 Len=0 TSval=226… |
| 72 | 6.418988995 | 10.0.2.4 | 10.0.2.5 | TCP | 66 | 36610 → 12345 [RST, ACK] Seq=14 Ack=13001 Win=78592 Len=0 TSva… |

**:V4**

בגרסה זו הלקוח מקבל כארגומנטים את פרטי השרת. לאחר מכן הוא פותח סוקט מסוג TCP ומתחבר לשרת בנתון.
הלקוח שולח את המחרוזת 'Hello, World!' לשרת 10 פעמים בכל הודעה. הוא מבצע 4 שליחות כאלה ולאחר מכן הוא מצפה לקבל מהשרת הודעה.
לאחר מכן הוא סוגר את החיבור עם השרת ומדפיס את המידע שהתקבל.

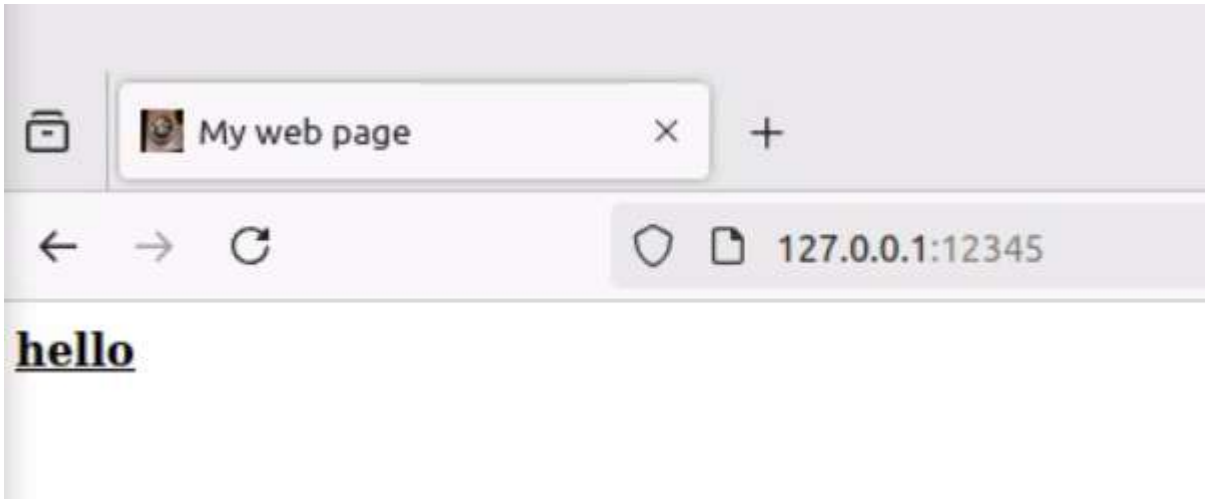
שרת - מגדירים את השרת כמו בV3 אז נסביר רק על הלולאה האינסופית:
מקבלים לקוח ומדפיסים את הכתובות של הלקוח (ip, port),
כעת בלולאה אינסופית: השרת ישן 5 שניות ואז מקבל מידע בגודל מקסימלי של 1024, אם הבאפר אז הוא עושה break אחרת הוא מדפיס אותו ושולח חזרה ללקוח באותיות גדולות. אחרי הלולאה הוא סוגר את החיבור.

הלקוח אכן שולח 4 הודעות לשרת אבל בשכבת הTCP המידע התאחד ל2 חבילות (בצהוב).
לאחר מכן השרת שולח את המידע שנשלח אליו בחזרה ללקוח באותיות גדולות בחבילה אחד (ירוק)
הלקוח סוגר את החיבור לאחר קריאת המידע מהשרת והשרת מאשר את הסגירה.

| No. | Time | Source | Destination | Protocc | Lenc | Info |
|---|---|---|---|---|---|---|
| 10 | 2.490323268 | 10.0.2.4 | 10.0.2.5 | TCP | 74 | 44482 → 12345 [SYN] Seq=0 Win=64240 Len |
| 11 | 2.490347630 | 10.0.2.5 | 10.0.2.4 | TCP | 74 | 12345 → 44482 [SYN, ACK] Seq=0 Ack=1 Wi |
| 12 | 2.504968570 | 10.0.2.4 | 10.0.2.5 | TCP | 66 | 44482 → 12345 [ACK] Seq=1 Ack=1 Win=642 |
| 13 | 2.504969033 | 10.0.2.4 | 10.0.2.5 | TCP | 196 | 44482 → 12345 [PSH, ACK] Seq=1 Ack=1 Wi |
| 14 | 2.505048794 | 10.0.2.5 | 10.0.2.4 | TCP | 66 | 12345 → 44482 [ACK] Seq=1 Ack=131 Win=6 |
| 15 | 2.505473867 | 10.0.2.4 | 10.0.2.5 | TCP | 456 | 44482 → 12345 [PSH, ACK] Seq=131 Ack=1 |
| 16 | 2.505483467 | 10.0.2.5 | 10.0.2.4 | TCP | 66 | 12345 → 44482 [ACK] Seq=1 Ack=521 Win=6 |
| 59 | 7.507413023 | 10.0.2.5 | 10.0.2.4 | TCP | 586 | 12345 → 44482 [PSH, ACK] Seq=1 Ack=521 |
| 60 | 7.510393909 | 10.0.2.4 | 10.0.2.5 | TCP | 66 | 44482 → 12345 [ACK] Seq=521 Ack=521 Win |
| 61 | 7.510394206 | 10.0.2.4 | 10.0.2.5 | TCP | 66 | 44482 → 12345 [FIN, ACK] Seq=521 Ack=52 |
| 64 | 7.551038363 | 10.0.2.5 | 10.0.2.4 | TCP | 66 | 12345 → 44482 [ACK] Seq=521 Ack=522 Win |

# חלק 2

## בקשה דרך הדפדפן:



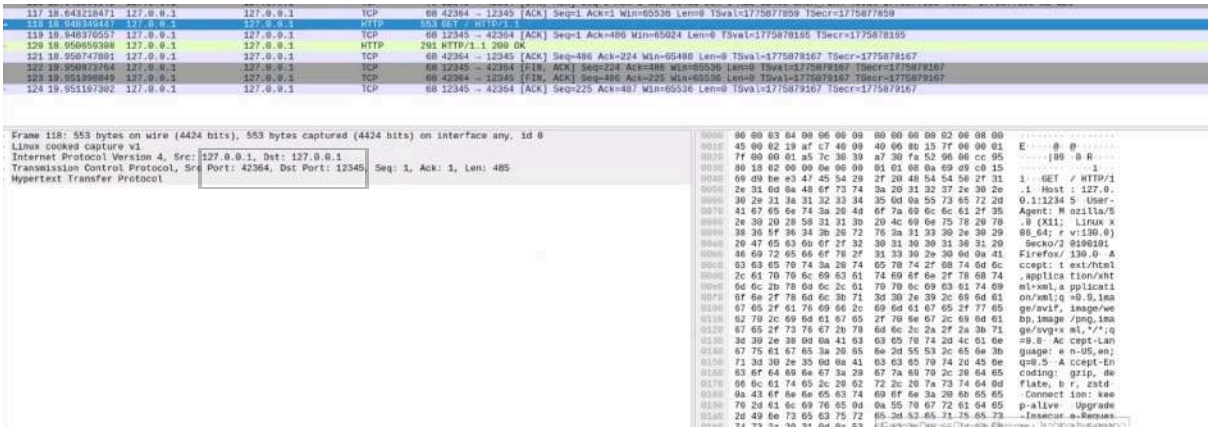

הדפסת השרת:

```
ofry@ofry-virtual-machine:~/Desktop/reshatim/ex2-networking$ python3 server.py 12345
GET / HTTP/1.1
Host: 127.0.0.1:12345
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:130.0) Gecko/20100101 Firefox/130.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/png,image/svg+xml,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br, zstd
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: none
Sec-Fetch-User: ?1
Priority: u=0, i
```

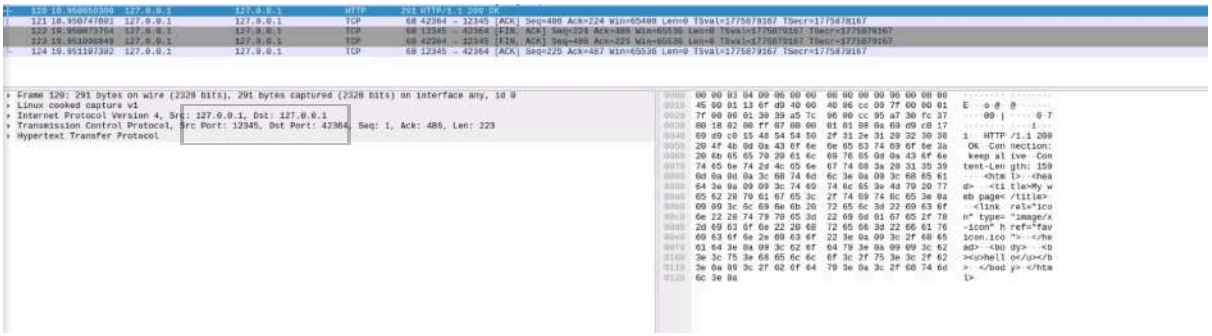
הסנפה:

| | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 115 | 18.643155480 | 127.0.0.1 | 127.0.0.1 | TCP | 76 | 42364 → 12345 [SYN] Seq=0 Win=65495 Len=0 MS |
| 116 | 18.643206642 | 127.0.0.1 | 127.0.0.1 | TCP | 76 | 12345 → 42364 [SYN, ACK] Seq=0 Ack=1 Win=654 |
| 117 | 18.643218471 | 127.0.0.1 | 127.0.0.1 | TCP | 68 | 42364 → 12345 [ACK] Seq=1 Ack=1 Win=65536 Le |
| 118 | 18.948349447 | 127.0.0.1 | 127.0.0.1 | HTTP | 553 | GET / HTTP/1.1 |
| 119 | 18.948370557 | 127.0.0.1 | 127.0.0.1 | TCP | 68 | 12345 → 42364 [ACK] Seq=1 Ack=486 Win=65024 |
| 120 | 18.950659308 | 127.0.0.1 | 127.0.0.1 | HTTP | 291 | HTTP/1.1 200 OK |
| 121 | 18.950747801 | 127.0.0.1 | 127.0.0.1 | TCP | 68 | 42364 → 12345 [ACK] Seq=486 Ack=224 Win=6540 |
| 122 | 19.950873764 | 127.0.0.1 | 127.0.0.1 | TCP | 68 | 12345 → 42364 [FIN, ACK] Seq=224 Ack=486 Win |
| 123 | 19.951098849 | 127.0.0.1 | 127.0.0.1 | TCP | 68 | 42364 → 12345 [FIN, ACK] Seq=486 Ack=225 Win |
| 124 | 19.951107302 | 127.0.0.1 | 127.0.0.1 | TCP | 68 | 12345 → 42364 [ACK] Seq=225 Ack=487 Win=6553 |

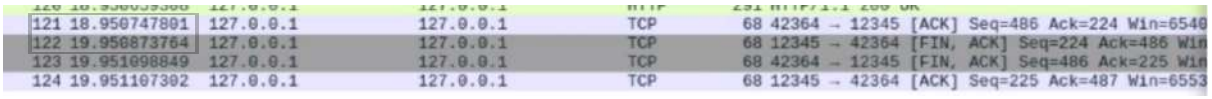
יש לחיצת ידיים ואז ואז הקליינט (הדפדפן) שולח בקשת http לשרת.



ואז השרת שולח תשובה בhttp:

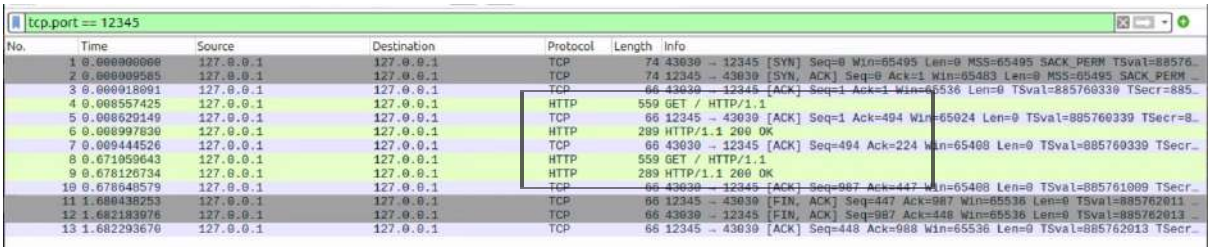


בקוד השרת הוגדר שהחיבור נסגר אחרי שניה שלא נשלחה בקשה, ואכן ניתן לראות שהשרת שולח fin אחרי שנייה שאין בקשה מהלקוח.



| | | | | | |
|---|---|---|---|---|---|
| 121 | 18.950747801 | 127.0.0.1 | 127.0.0.1 | TCP | 68 42364 → 12345 [ACK] Seq=486 Ack=224 Win=6540 |
| 122 | 19.950873764 | 127.0.0.1 | 127.0.0.1 | TCP | 68 12345 → 42364 [FIN, ACK] Seq=224 Ack=486 Win |
| 123 | 19.951098849 | 127.0.0.1 | 127.0.0.1 | TCP | 68 42364 → 12345 [FIN, ACK] Seq=486 Ack=225 Win |
| 124 | 19.951107302 | 127.0.0.1 | 127.0.0.1 | TCP | 68 12345 → 42364 [ACK] Seq=225 Ack=487 Win=6553 |

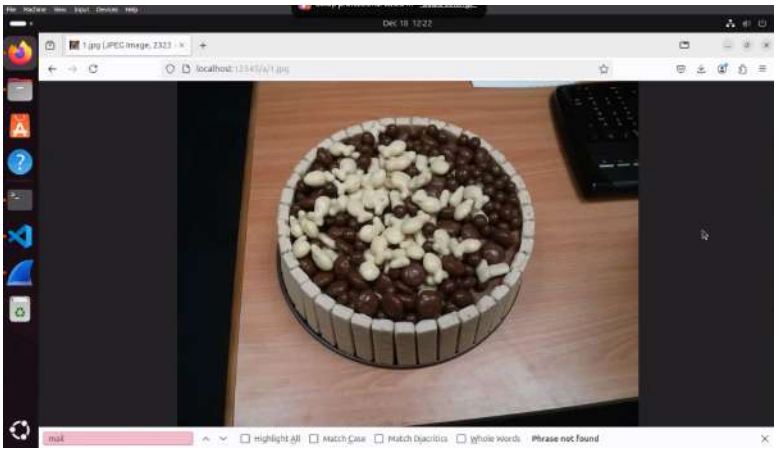
כעת הרצנו את אותה בקשה פעם אחרי פעם בהפרש של פחות משניה, ניתן לראות שמגיעות 2 בקשות באותו החיבור.



tcp.port == 12345

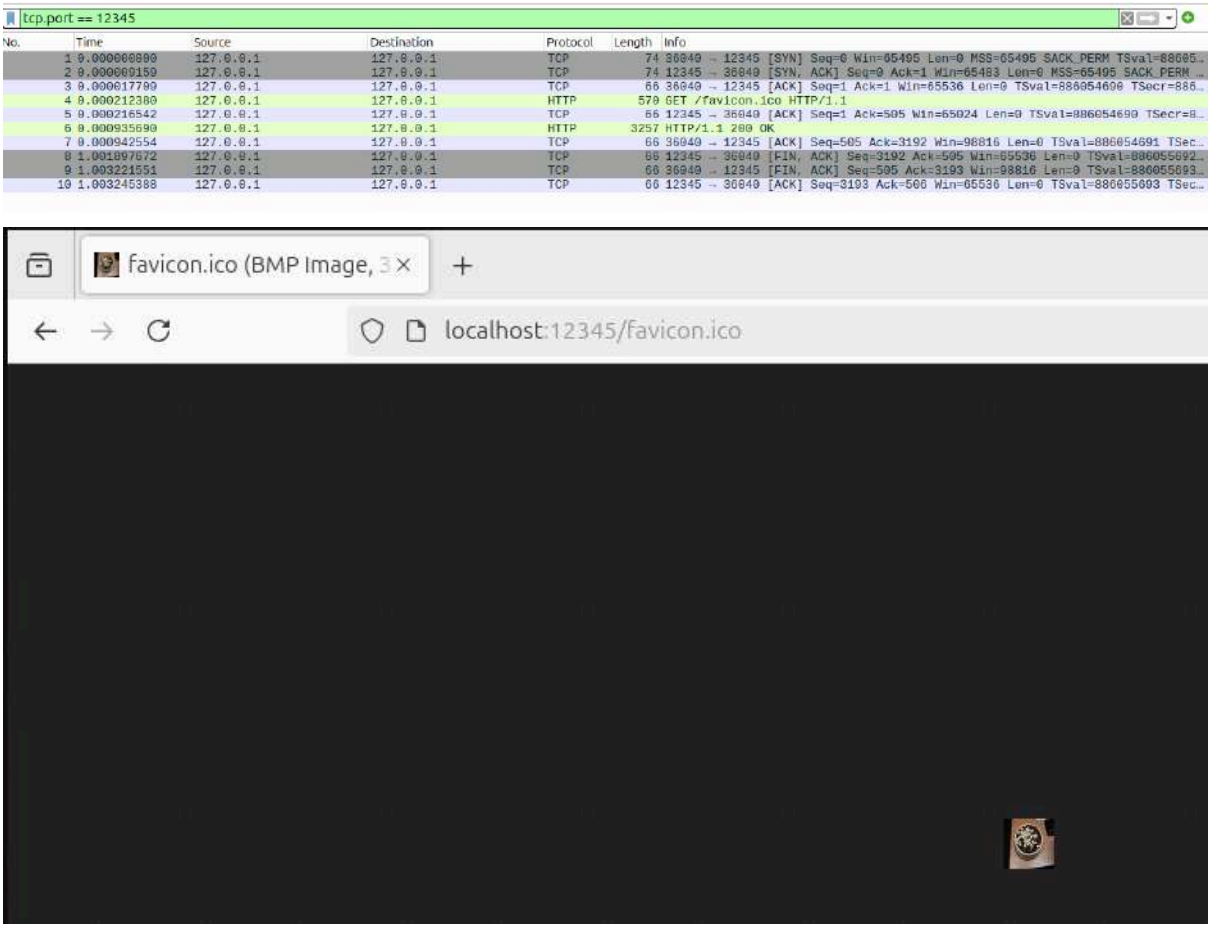
| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 1 | 0.000000000 | 127.0.0.1 | 127.0.0.1 | TCP | 74 | 43030 → 12345 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM TSval=88576… |
| 2 | 0.000009585 | 127.0.0.1 | 127.0.0.1 | TCP | 74 | 12345 → 43030 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM … |
| 3 | 0.000018091 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 43030 → 12345 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=885760330 TSecr=885… |
| 4 | 0.008557425 | 127.0.0.1 | 127.0.0.1 | HTTP | 559 | GET / HTTP/1.1 |
| 5 | 0.008629149 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 12345 → 43039 [ACK] Seq=1 Ack=494 Win=65024 Len=0 TSval=885760339 TSecr=8… |
| 6 | 0.008997830 | 127.0.0.1 | 127.0.0.1 | HTTP | 289 | HTTP/1.1 200 OK |
| 7 | 0.009444526 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 43030 → 12345 [ACK] Seq=494 Ack=224 Win=65408 Len=0 TSval=885760339 TSecr… |
| 8 | 0.671059643 | 127.0.0.1 | 127.0.0.1 | HTTP | 559 | GET / HTTP/1.1 |
| 9 | 0.678126734 | 127.0.0.1 | 127.0.0.1 | HTTP | 289 | HTTP/1.1 200 OK |
| 10 | 0.678648579 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 43030 → 12345 [ACK] Seq=987 Ack=447 Win=65408 Len=0 TSval=885761009 TSecr… |
| 11 | 1.680438253 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 12345 → 43030 [FIN, ACK] Seq=447 Ack=987 Win=65536 Len=0 TSval=885762011 … |
| 12 | 1.682183976 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 43030 → 12345 [FIN, ACK] Seq=987 Ack=448 Win=65536 Len=0 TSval=885762013 … |
| 13 | 1.682293670 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 12345 → 43030 [ACK] Seq=448 Ack=988 Win=65536 Len=0 TSval=885762013 TSecr… |

דוגמה לבקשה של תמונה:



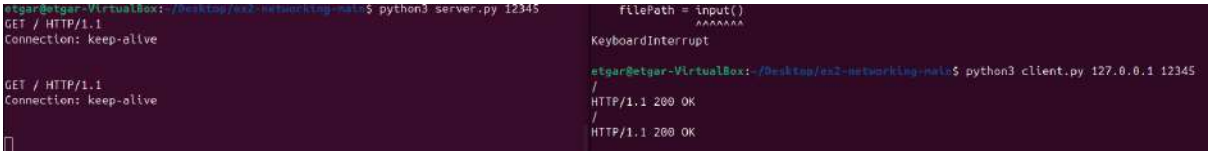

דוגמא להסנפת אייקון:

יש לנו לחיצת ידיים, לאחר מכן בקשה מהדפדפן של הקובץ. השרת לאחר מכן שולח את הקובץ והחיבור נסגר לאחר שנייה של חוסר שימוש.

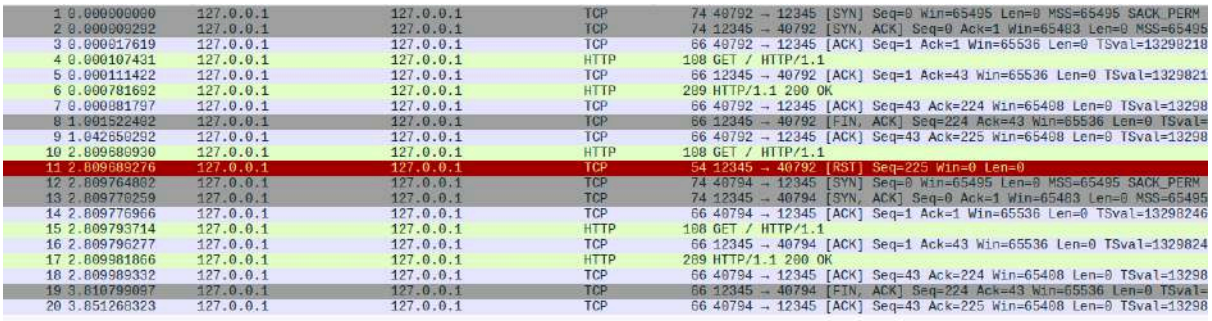



| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 1 | 0.000000000 | 127.0.0.1 | 127.0.0.1 | TCP | 74 | 36040 → 12345 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM TSval=88605… |
| 2 | 0.000009159 | 127.0.0.1 | 127.0.0.1 | TCP | 74 | 12345 → 36040 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM … |
| 3 | 0.000017799 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 36040 → 12345 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=886054690 TSecr=886… |
| 4 | 0.000212380 | 127.0.0.1 | 127.0.0.1 | HTTP | 570 | GET /favicon.ico HTTP/1.1 |
| 5 | 0.000216542 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 12345 → 36040 [ACK] Seq=1 Ack=505 Win=65024 Len=0 TSval=886054690 TSecr=8… |
| 6 | 0.000935690 | 127.0.0.1 | 127.0.0.1 | HTTP | 3257 | HTTP/1.1 200 OK |
| 7 | 0.000942554 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 36040 → 12345 [ACK] Seq=505 Ack=3192 Win=98816 Len=0 TSval=886054691 TSec… |
| 8 | 1.001897672 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 12345 → 36040 [FIN, ACK] Seq=3192 Ack=505 Win=65536 Len=0 TSval=886055692… |
| 9 | 1.003221551 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 36040 → 12345 [FIN, ACK] Seq=505 Ack=3193 Win=98816 Len=0 TSval=886055693… |
| 10 | 1.003245388 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 12345 → 36040 [ACK] Seq=3193 Ack=506 Win=65536 Len=0 TSval=886055693 TSec… |

# בקשה דרך הלקוח שלנו (המלך):



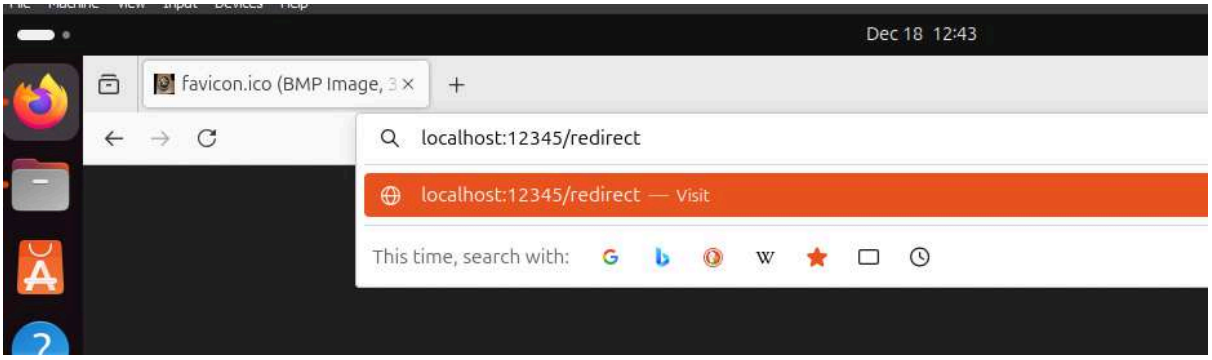

הלקוח שלנו שולח בקשה לקובץ index.html ומקבל תשובה מהשרת. השרת סוגר את החיבור כי עברה שניה ולכן כאשר הלקוח שולח בקשה שניה מתקבל RST. לאחר מכן הלקוח מבקש לפתוח חיבור חדש ושולח את הבקשה בשנית.

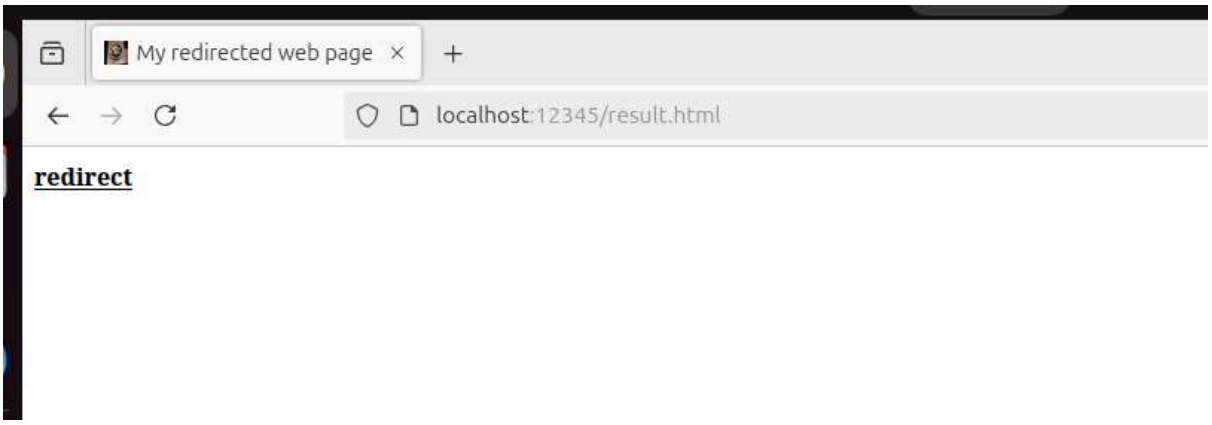


| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | 0.000000000 | 127.0.0.1 | 127.0.0.1 | TCP | 74 | 40792 → 12345 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM |
| 2 | 0.000009292 | 127.0.0.1 | 127.0.0.1 | TCP | 74 | 12345 → 40792 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 |
| 3 | 0.000017619 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 40792 → 12345 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=13298218 |
| 4 | 0.000107431 | 127.0.0.1 | 127.0.0.1 | HTTP | 108 | GET / HTTP/1.1 |
| 5 | 0.000111422 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 12345 → 40792 [ACK] Seq=1 Ack=43 Win=65536 Len=0 TSval=1329821 |
| 6 | 0.000781692 | 127.0.0.1 | 127.0.0.1 | HTTP | 289 | HTTP/1.1 200 OK |
| 7 | 0.000881797 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 40792 → 12345 [ACK] Seq=43 Ack=224 Win=65408 Len=0 TSval=13298 |
| 8 | 1.001522402 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 12345 → 40792 [FIN, ACK] Seq=224 Ack=43 Win=65536 Len=0 TSval= |
| 9 | 1.042650292 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 40792 → 12345 [ACK] Seq=43 Ack=225 Win=65408 Len=0 TSval=13298 |
| 10 | 2.809580930 | 127.0.0.1 | 127.0.0.1 | HTTP | 108 | GET / HTTP/1.1 |
| 11 | 2.809689276 | 127.0.0.1 | 127.0.0.1 | TCP | 54 | 12345 → 40792 [RST] Seq=225 Win=0 Len=0 |
| 12 | 2.809764802 | 127.0.0.1 | 127.0.0.1 | TCP | 74 | 40794 → 12345 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM |
| 13 | 2.809770259 | 127.0.0.1 | 127.0.0.1 | TCP | 74 | 12345 → 40794 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 |
| 14 | 2.809776966 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 40794 → 12345 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=13298246 |
| 15 | 2.809793714 | 127.0.0.1 | 127.0.0.1 | HTTP | 108 | GET / HTTP/1.1 |
| 16 | 2.809796277 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 12345 → 40794 [ACK] Seq=1 Ack=43 Win=65536 Len=0 TSval=1329824 |
| 17 | 2.809981866 | 127.0.0.1 | 127.0.0.1 | HTTP | 289 | HTTP/1.1 200 OK |
| 18 | 2.809989332 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 40794 → 12345 [ACK] Seq=43 Ack=224 Win=65408 Len=0 TSval=13298 |
| 19 | 3.810799097 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 12345 → 40794 [FIN, ACK] Seq=224 Ack=43 Win=65536 Len=0 TSval= |
| 20 | 3.851268323 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 40794 → 12345 [ACK] Seq=43 Ack=225 Win=65408 Len=0 TSval=13298 |

## <u>כעת נראה הסנפה של בקשת redirect:</u>

## מהדפדפן:




ניתן לראות שקיבלנו מהשרת קובץ של result.html על בקשת הredirect.

## מהלקוח:

הדפסת השרת:

```
GET /redirect HTTP/1.1
Connection: keep-alive


GET /result.html HTTP/1.1
Connection: keep-alive
```

הדפסת הלקוח:

```
etgar@etgar-VirtualBox:~/Desktop/ex2-networking-main$ python3 client.py 127.0.0.1 12345
/redirect
HTTP/1.1 301 Moved Permanently
HTTP/1.1 200 OK
```

כאן ניתן לראות שהלקוח שולח בקשה לredirect ומקבל תשובה עם מיקום הקובץ והחיבור נסגר כחלק מהפרוטוקול שהוגדר בתרגיל. הלקוח לאחר מכן שולח בקשה חדשה עם מיקום הקובץ שקיבל בתשובת השרת ומקבל את הקובץ result.html.

| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | 0.000000000 | 127.0.0.1 | 127.0.0.1 | TCP | 74 | 53530 → 12345 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM TSval |
| 2 | 0.000011302 | 127.0.0.1 | 127.0.0.1 | TCP | 74 | 12345 → 53530 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_ |
| 3 | 0.000022863 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 53530 → 12345 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=1330103961 TS |
| 4 | 0.000058608 | 127.0.0.1 | 127.0.0.1 | HTTP | 116 | GET /redirect HTTP/1.1 |
| 5 | 0.000062572 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 12345 → 53530 [ACK] Seq=1 Ack=51 Win=65536 Len=0 TSval=1330103961 T |
| 6 | 0.001041017 | 127.0.0.1 | 127.0.0.1 | TCP | 143 | 12345 → 53530 [PSH, ACK] Seq=1 Ack=51 Win=65536 Len=77 TSval=133010 |
| 7 | 0.001147031 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 53530 → 12345 [ACK] Seq=51 Ack=78 Win=65536 Len=0 TSval=1330103962 |
| 8 | 0.001169822 | 127.0.0.1 | 127.0.0.1 | HTTP | 66 | HTTP/1.1 301 Moved Permanently |
| 9 | 0.001707774 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 53530 → 12345 [FIN, ACK] Seq=51 Ack=79 Win=65536 Len=0 TSval=133010 |
| 10 | 0.001785790 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 12345 → 53530 [ACK] Seq=79 Ack=52 Win=65536 Len=0 TSval=1330103963 |
| 11 | 0.001835929 | 127.0.0.1 | 127.0.0.1 | TCP | 74 | 53538 → 12345 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM TSval |
| 12 | 0.001842688 | 127.0.0.1 | 127.0.0.1 | TCP | 74 | 12345 → 53538 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_ |
| 13 | 0.001848875 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 53538 → 12345 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=1330103963 TS |
| 14 | 0.001970435 | 127.0.0.1 | 127.0.0.1 | HTTP | 119 | GET /result.html HTTP/1.1 |
| 15 | 0.001974839 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 12345 → 53538 [ACK] Seq=1 Ack=54 Win=65536 Len=0 TSval=1330103963 T |
| 16 | 0.004454001 | 127.0.0.1 | 127.0.0.1 | HTTP | 244 | HTTP/1.1 200 OK |
| 17 | 0.004710199 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 53538 → 12345 [ACK] Seq=54 Ack=179 Win=65408 Len=0 TSval=1330103966 |

## <u>כעת נראה הסנפה של בקשת קובץ לא קיים:</u>

### מהלקוח:

```
$ python3 client.py 127.0.0.1 12346
/what
HTTP/1.1 404 Not Found
```

### מהשרת:

| | | | | | | |
|---|---|---|---|---|---|---|
| 5 | 0.755189599 | 127.0.0.1 | 127.0.0.1 | TCP | 74 | 43268 → 12346 [SYN] Seq=0 Win=65495 Len= |
| 6 | 0.755198368 | 127.0.0.1 | 127.0.0.1 | TCP | 74 | 12346 → 43268 [SYN, ACK] Seq=0 Ack=1 Win |
| 7 | 0.755206781 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 43268 → 12346 [ACK] Seq=1 Ack=1 Win=6553 |
| 8 | 0.755230513 | 127.0.0.1 | 127.0.0.1 | HTTP | 112 | GET /what HTTP/1.1 |
| 9 | 0.755233478 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 12346 → 43268 [ACK] Seq=1 Ack=47 Win=655 |
| 10 | 0.755609490 | 127.0.0.1 | 127.0.0.1 | TCP | 111 | 12346 → 43268 [PSH, ACK] Seq=1 Ack=47 Wi |
| 11 | 0.755909605 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 43268 → 12346 [ACK] Seq=47 Ack=46 Win=65 |
| 12 | 0.756115336 | 127.0.0.1 | 127.0.0.1 | HTTP | 66 | HTTP/1.1 404 Not Found |
| 13 | 0.756388381 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 43268 → 12346 [FIN, ACK] Seq=47 Ack=47 W |
| 14 | 0.756398442 | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 12346 → 43268 [ACK] Seq=47 Ack=48 Win=65 |