```
noatzur@noatzur:~/Desktop/Labsetup$ telnet 10.0.6.24
Trying 10.0.6.24...
```

כאן ניתן לראות שהפסו את הפקטות TCP
בטווח של 100-10:

```
noatzur@noatzur: ~/Desktop/Labsetup
noatzur@noatzur:~/Desktop/Labsetup$ sudo ./sniffer
[sudo] password for noatzur:
       From: 10.0.2.15
         To: 10.0.6.24
im an TCP packet !!!

       From: 10.0.2.15
         To: 10.0.6.24
im an TCP packet !!!

       From: 10.0.2.15
         To: 10.0.6.24
im an TCP packet !!!

       From: 10.0.2.15
         To: 10.0.6.24
im an TCP packet !!!

       From: 10.0.2.15
         To: 34.107.221.82
im an TCP packet !!!

       From: 10.0.2.15
         To: 34.107.221.82
im an TCP packet !!!

       From: 10.0.2.15
         To: 34.107.221.82
im an TCP packet !!!

       From: 10.0.2.15
```

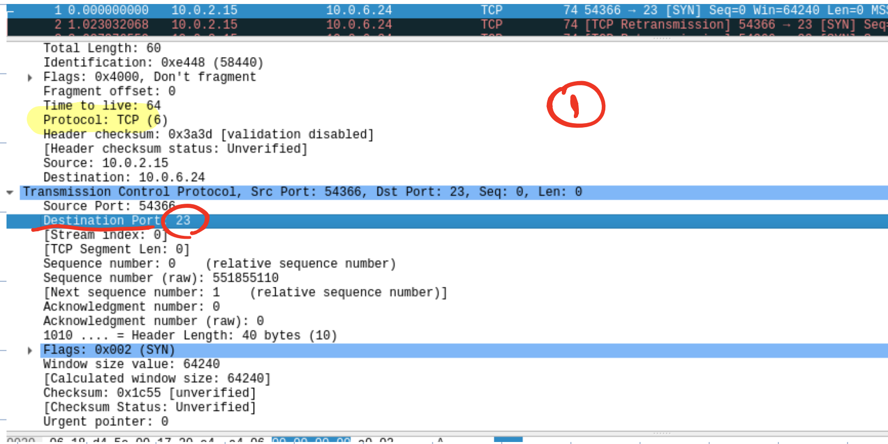
① ניתן לראות כי שהוא נשלח לפורט 23

② ניתן לראות כי שהוא נשלח לפורט 80.

שניהם בטווח.

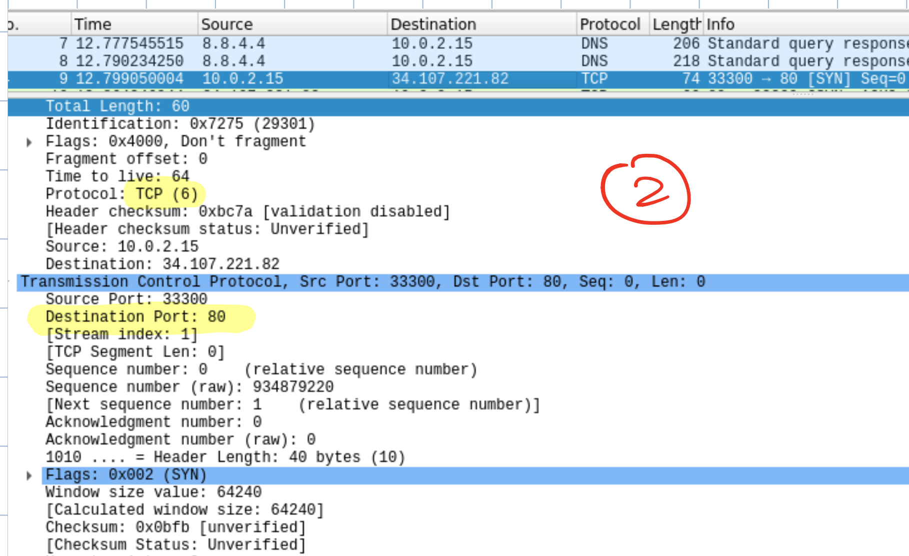
Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	10.0.2.15	10.0.6.24	TCP	74	54366 → 23 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 T…
2	1.023032068	10.0.2.15	10.0.6.24	TCP	74	[TCP Retransmission] 54366 → 23 [SYN] Seq=0 Win=64240 Len=0 M…
3	3.037376559	10.0.2.15	10.0.6.24	TCP	74	[TCP Retransmission] 54366 → 23 [SYN] Seq=0 Win=64240 Len=0 M…
4	7.229806345	10.0.2.15	10.0.6.24	TCP	74	[TCP Retransmission] 54366 → 23 [SYN] Seq=0 Win=64240 Len=0 M…
5	12.709770056	10.0.2.15	8.8.4.4	DNS	95	Standard query 0xe012 A detectportal.firefox.com OPT
6	12.710085976	10.0.2.15	8.8.4.4	DNS	95	Standard query 0x34c6 AAAA detectportal.firefox.com OPT
7	12.777545515	8.8.4.4	10.0.2.15	DNS	206	Standard query response 0xe012 A detectportal.firefox.com CNA…
8	12.790234250	8.8.4.4	10.0.2.15	DNS	218	Standard query response 0x34c6 AAAA detectportal.firefox.com …
9	12.799050000	10.0.2.15	34.107.221.82	TCP	74	33300 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 T…
10	12.864246944	34.107.221.82	10.0.2.15	TCP	60	80 → 33300 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
11	12.864356038	10.0.2.15	34.107.221.82	TCP	54	33300 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
12	12.864549563	10.0.2.15	34.107.221.82	HTTP	350	GET /success.txt HTTP/1.1
13	12.865268073	34.107.221.82	10.0.2.15	TCP	60	80 → 33300 [ACK] Seq=1 Ack=297 Win=65535 Len=0
14	12.932918025	34.107.221.82	10.0.2.15	HTTP	274	HTTP/1.1 200 OK (text/plain)
15	12.932994138	10.0.2.15	34.107.221.82	TCP	54	33300 → 80 [ACK] Seq=297 Ack=221 Win=64020 Len=0
16	12.998538915	10.0.2.15	8.8.4.4	DNS	82	Standard query 0xa412 A example.org OPT
17	13.012384684	10.0.2.15	8.8.4.4	DNS	82	Standard query 0xefd0 AAAA example.org OPT
18	13.012968541	10.0.2.15	8.8.4.4	DNS	84	Standard query 0x2bd4 A ipv4only.arpa OPT
19	13.013694636	10.0.2.15	8.8.4.4	DNS	84	Standard query 0x7ba6 AAAA ipv4only.arpa OPT
20	13.018650942	10.0.2.15	8.8.4.4	DNS	106	Standard query 0xaed6 A content-signature-2.cdn.mozilla.net O…
21	13.018906963	10.0.2.15	8.8.4.4	DNS	106	Standard query 0xc1cf AAAA content-signature-2.cdn.mozilla.ne…
22	13.024058936	10.0.2.15	34.107.221.82	TCP	74	33302 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 T…



```
1 0.000000000   10.0.2.15   10.0.6.24   TCP   74 54366 → 23 [SYN] Seq=0 Win=64240 Len=0 MSS
2 1.023032068   10.0.2.15   10.0.6.24   TCP   74 [TCP Retransmission] 54366 → 23 [SYN] Seq=
    Total Length: 60
    Identification: 0xe448 (58440)
  ▸ Flags: 0x4000, Don't fragment
    Fragment offset: 0
    Time to live: 64
    Protocol: TCP (6)
    Header checksum: 0x3a3d [validation disabled]
    [Header checksum status: Unverified]
    Source: 10.0.2.15
    Destination: 10.0.6.24
▾ Transmission Control Protocol, Src Port: 54366, Dst Port: 23, Seq: 0, Len: 0
    Source Port: 54366
    Destination Port: 23
    [Stream index: 0]
    [TCP Segment Len: 0]
    Sequence number: 0    (relative sequence number)
    Sequence number (raw): 551855110
    [Next sequence number: 1    (relative sequence number)]
    Acknowledgment number: 0
    Acknowledgment number (raw): 0
    1010 .... = Header Length: 40 bytes (10)
  ▸ Flags: 0x002 (SYN)
    Window size value: 64240
    [Calculated window size: 64240]
    Checksum: 0x1c55 [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
```

1



No.	Time	Source	Destination	Protocol	Length	Info
7	12.777545515	8.8.4.4	10.0.2.15	DNS	206	Standard query response
8	12.790234250	8.8.4.4	10.0.2.15	DNS	218	Standard query response
9	12.799050004	10.0.2.15	34.107.221.82	TCP	74	33300 → 80 [SYN] Seq=0

```
    Total Length: 60
    Identification: 0x7275 (29301)
  ▸ Flags: 0x4000, Don't fragment
    Fragment offset: 0
    Time to live: 64
    Protocol: TCP (6)
    Header checksum: 0xbc7a [validation disabled]
    [Header checksum status: Unverified]
    Source: 10.0.2.15
    Destination: 34.107.221.82
  Transmission Control Protocol, Src Port: 33300, Dst Port: 80, Seq: 0, Len: 0
    Source Port: 33300
    Destination Port: 80
    [Stream index: 1]
    [TCP Segment Len: 0]
    Sequence number: 0    (relative sequence number)
    Sequence number (raw): 934879220
    [Next sequence number: 1    (relative sequence number)]
    Acknowledgment number: 0
    Acknowledgment number (raw): 0
    1010 .... = Header Length: 40 bytes (10)
  ▸ Flags: 0x002 (SYN)
    Window size value: 64240
    [Calculated window size: 64240]
    Checksum: 0x0bfb [unverified]
    [Checksum Status: Unverified]
```

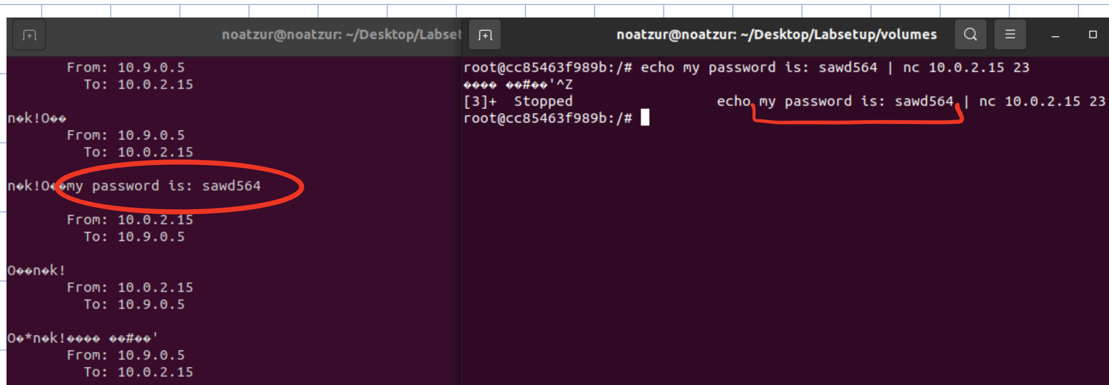
2

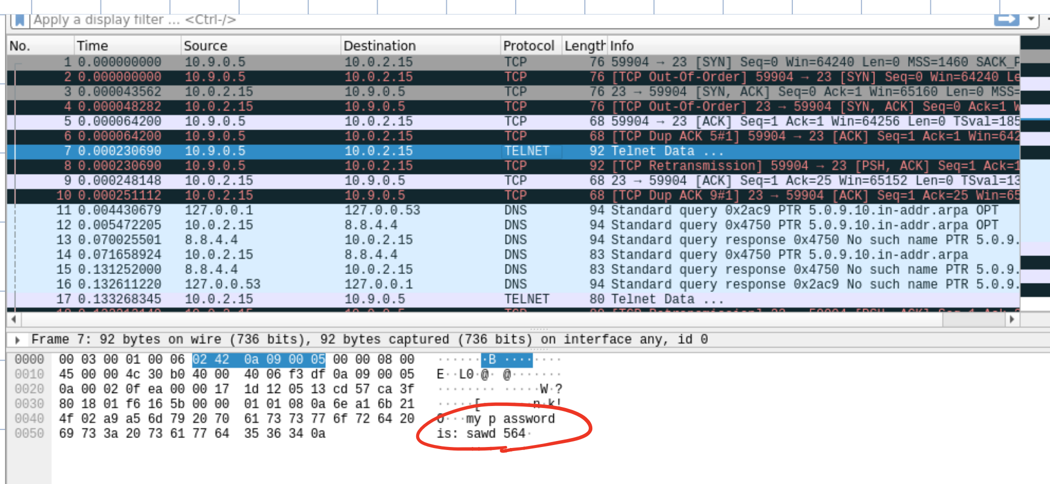
2.1c:

בסעיף זה התבקשנו להדפיס סיסמא של user אשר משתמש בפרוטוקול telnet, פרוטוקול זה משתמש ב port 23 ואינו מאובטח ולכן ניתן לראות את ה-data שהמשתמש מזין.

בתמונה מטה הדפסתי למסך את כל ה-data שהתקבל והסיסמא מסומנת בצורה ידנית.

בנוסף צירפנו גם תמונה של WS שם ניתן לראות את הdata. 8 מנות נשלחו ע"י telnet



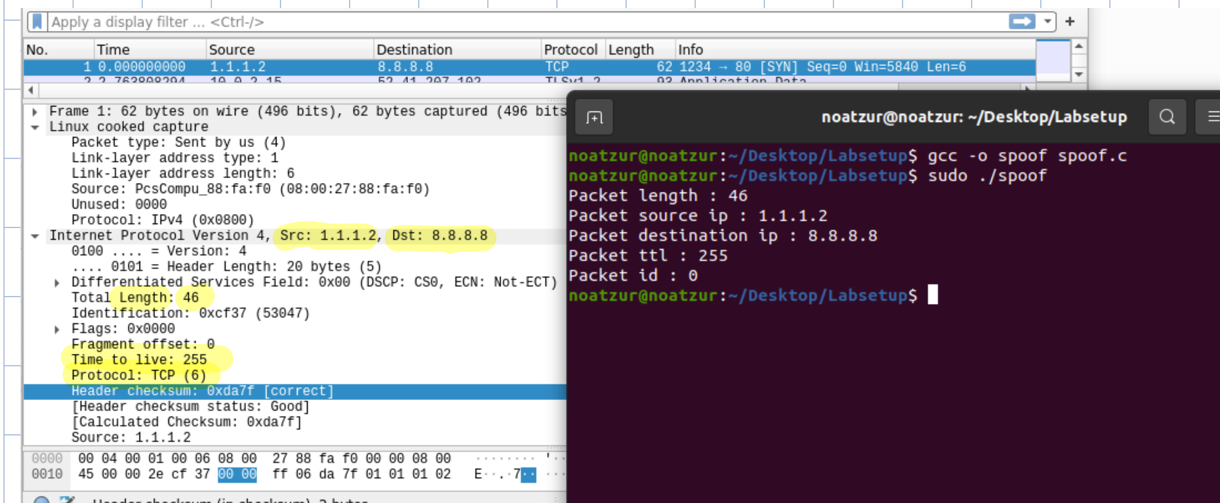


2.2A :

Spoof קובץ spoof.c

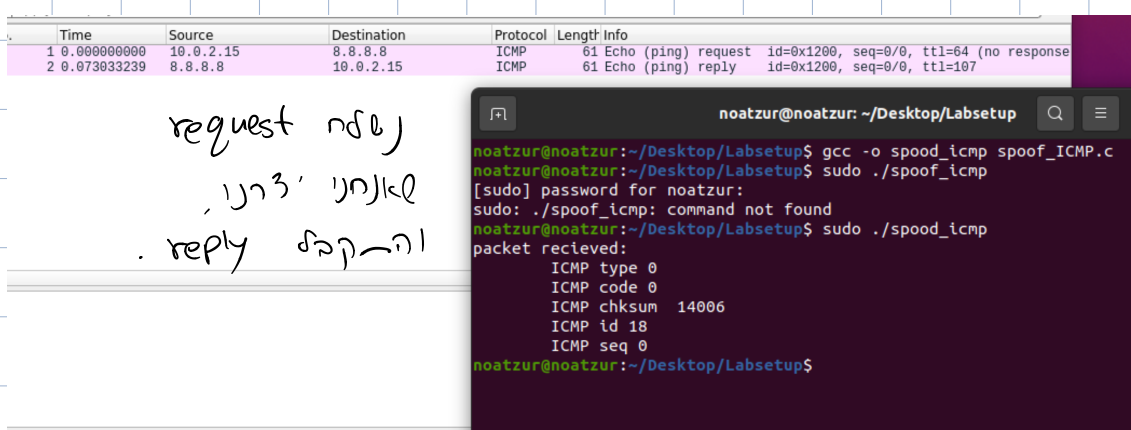
בסעיף זה התבקשנו לכתוב את תוכנית הSpoof שלנו.
כתבנו תוכנית שיוצרת פקטה מסוג TCP, שהיעד שלה הוא
8.8.8.8 וה IP source שלה 1.1.1.2 (IP שלא קיים בכלל...)

בתמונה המצורפת ניתן לראות את הפקטה שיצרנו ואת פרטיה.
וב WS ניתן לראות שאכן נשלחה הפקטה הזו.



2.2B: Spoof ICMP request   קובץ spoof_ICMP.c

בסעיף זה התבקשנו לכתוב תכנית spoof לייצור פקטה מסוג ICMP request (id=8), השתמשנו בקוד שכתבנו בשאלה הקודמת כי הוא עונה על תנאים אלו בדיוק.

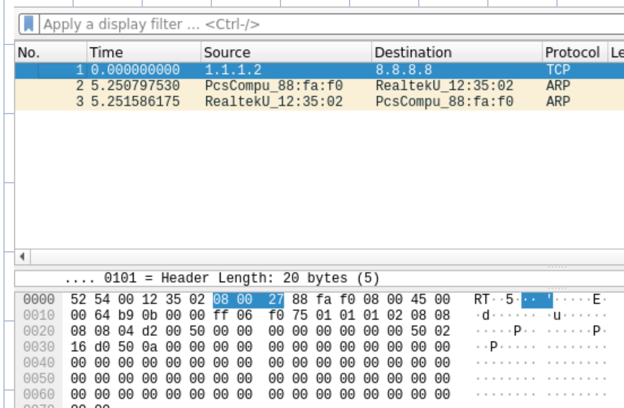


נשלחה request שאנחנו יצרנו, והתקבל reply.

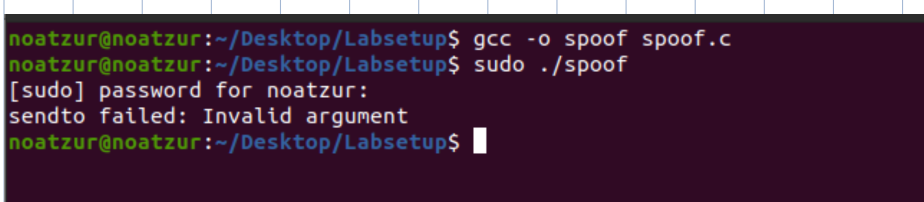
בנוסף בסעיף זה התבקשנו לענות על 3 שאלות:

Question 4:

לאחר ניסוי קצר שעשינו בקובץ spoof.c, שבו שינינו את גודל
ה ip header לגלות כי לא משנה מה הגודל שנגדיר לו, כל עוד
הערך גדול מ-20 כי זה הגודל המינימלי.
(וכן לשים לב גם ערכים גדולים, למשל 100, הפקטה נשלחה באופן תקין
ו"מרפד" את ה data ב-אפסים



ריפוד באפסים ←
כאשר הגדרנו את הערך
ל-100.



השגיאה שנוצרה
כאשר הגדרנו
את הגודל
להיות 10.

Question 5:

גם בסעיף זה בדקנו ניסוי קטן כדי לבחון את
מה קורה אם השורה שמחשבת checksum עבור השדה המתאים ב Iph
ושמנו רק 0, שליחת הפקטה עברה באופן תקין, וגם ב ws ניתן
היה לראות שה header checksum הוא correct וה status הוא
good. מכאן הבנו שה raw socket מחשב עבורנו את ה checksum
ולכן אין הכרח שאנו נעשה את החישוב.

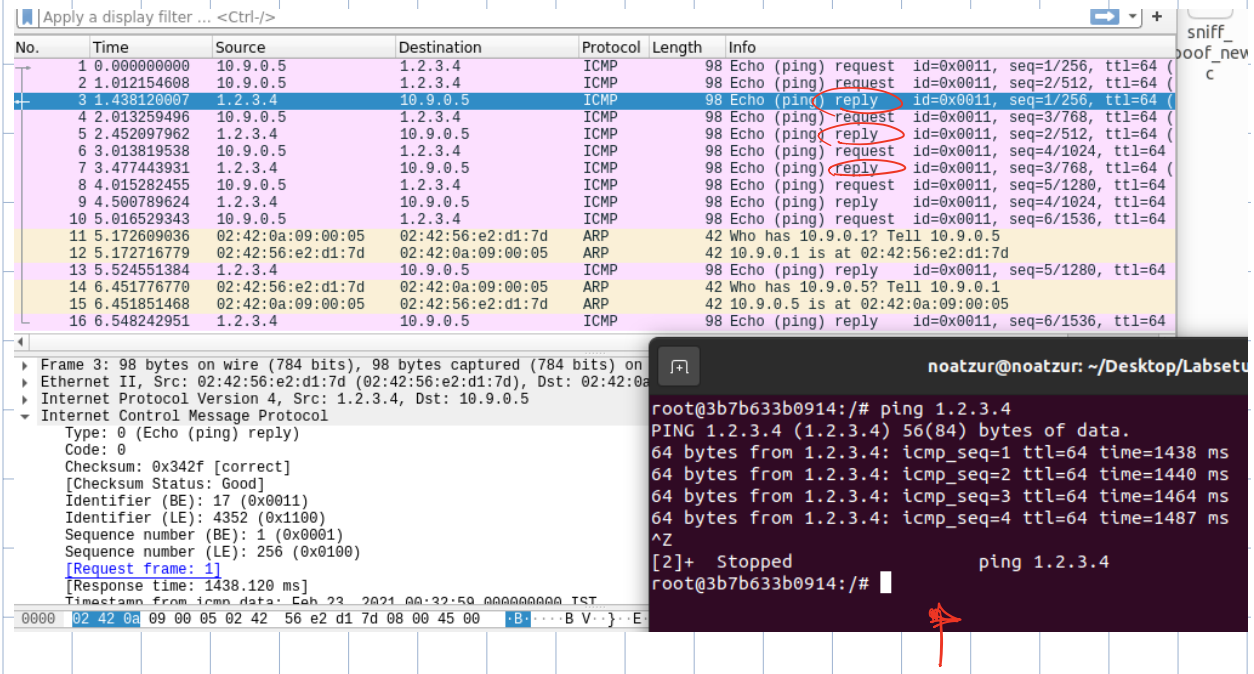
Question 6:

על מנת שנוכל לייצר פאקטה משלנו, לבחור ולשנות שדות ב-headers נצטרך הרשאות מנהל, לא כל משתמש רגיל יכול לעשות שינויים כאלה. כמו כן, גם כדי להסניף פאקטות שלא מיועדות לכרטיס הרשת הספציפי דרכו אנו עובדים, נצטרך root privilege. הפונ' של ספריית pcap דורשות זאת כמובן.
על מנת לבצע את כל האמור למעלה, התוכנית שננסה להריץ ללא הרשאה מתאימה לא תרוץ ותזרוק שגיאת Permission כך שלא נוכל לייצר פאקטות ובכך לעשות את עיקר התוכנית - Spoof.

2.3: Sniff & spoof

sniff_spoof_new.c

בסעיף זה התבקשנו "לחבר" את תוכנית ה Sniff שכתבנו עם תוכנית ה Spoof.
ב container של המשתמש הרגיל שלחנו ping ל IP שלא קיים 1.2.3.4 כאשר במקביל, ב container של ה attacker הרצנו את ה sniff & spoof שלנו.
התוכנית, ע"י פונ' ה Sniff שכתבנו, זיהתה את ה ping שנשלח מהמשתמש הרגיל (ICMP) וייצרה פאקטה reply ICMP מזויפת ושלחה לאותו IP ממנו התקבל ה ping.
כך בעצם "יצרנו מצב" שבו למשתמש הרגיל האומר שהוא קיבל תשובה מ IP שלא קיים, כאילו היה קיים...



ראה ש 1.2.3.4 אין קיים ומתקבל reply.