

## רשתות – תרגיל 2

שם : אליעד ארזואן

תז: 206482622

הקדמה והערות:

- עשינו באמצעות פרוטוקול UDP שרת המתנהג כמו פרוטוקול DNS.
- מכיוון שאנחנו עובדים על שרת מקומי רק שכתובת ה ip שלו היא 127.0.0.1 נתקשר עם כל שרת בפורט שונה ובאותו ip כי מדובר באותו מחשב. בהתחלה פירטתי יותר ואחרי זה יותר בקצרה כי זה חוזר על עצמו.
- יש לנו היררכיה בין שרתים:  
במקרים שאדגים:
  - Client – במקרה שלנו יהיה בפורט רנדומלי
  - Local server – יהיה בפורט 12345
  - Root server – יהיה בפורט 12346
  - Tld – במקרה שלנו ns.com – יהיה בפורט 12347
  - Auth – במקרה שלנו ns.facebook.com – יהיה בפורט 12348
- local server משמש כריזולבר המפנה אותנו לשרתים השונים.
- השרת מקבל פרמטרים למיין:  
X y z - כאשר x הוא מספר 1- אם השרת ריזולבר 0- אם לא  
y- הפורט בו יעבוד השרת מעליו בהיררכיה. z- שם קובץ המיפויים
- אני אסביר בחבילה אחת במורחב איך היא בנויה לפי שכבות ובשאר יותר בקצרה.
- שמרתי בקובץ המיפויים גם פורט עבור השרתים כי אנחנו עובדים בפורטים.
- תווים מפרידים בשליחה: | יפריד בין ההודעה שנשלח לשרתים שאינם ריסולברים לבין "תזכורת" למה שהם צריכים לעשות. @ יפריד בין ההודעה לתיאור של האובייקט של המיפוי.

נריץ את התוכנית שכתבנו במספר תרחישים:

**לקוח יחיד מול שרת בודד**

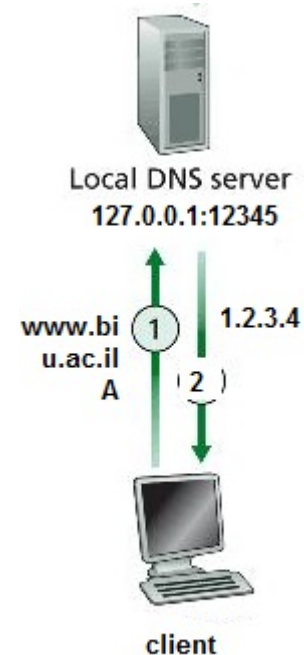
כזכור השרת והלקוח בכתובת 127.0.0.1 מה שמשנה זה הפורטים.

הפורט של השרת הוא 12345 ושל הלקוח במקרה זה 51053.

מכיוון שהלקוח מול שרת בודד וזה לא תרחיש של איטרציה התשובה תהיה בהכרח מקובץ המיפויים של השרת (אחרת מדובר על המקרים הבאים שצריך לבדוק)

קובץ המיפויים של השרת מכיל את התשובה לשאלתה A `www.biu.ac.il`

הבקשה תראה בצורה הבאה:



**בצד הלקוח:**

נבקש שאילתת A מהשרת עבור `www.biu.ac.il`.

```
Run: server client
C:\Python27\python.exe "C:/Users/eliad1998/Documents/2
Message to send: www.biu.ac.il A
Server sent: The answer for www.biu.ac.il is 1.2.3.4
Message to send:
```

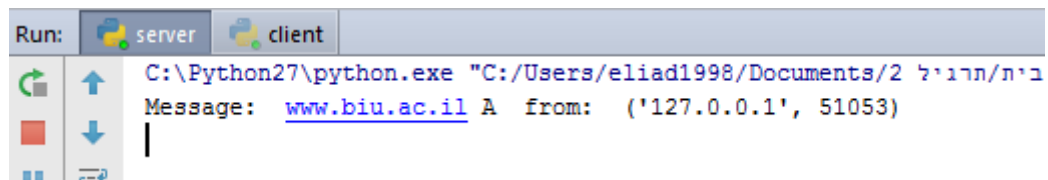
שלחנו בקשת A ואנחנו מצפים לקבל תשובה מהשרת.

קבלנו תשובה: הכתובת ip של הדומיין היא 1.2.3.4.

קבלנו תשובה מהרשומות של השרת כי הדומיין ברשומות שלו.

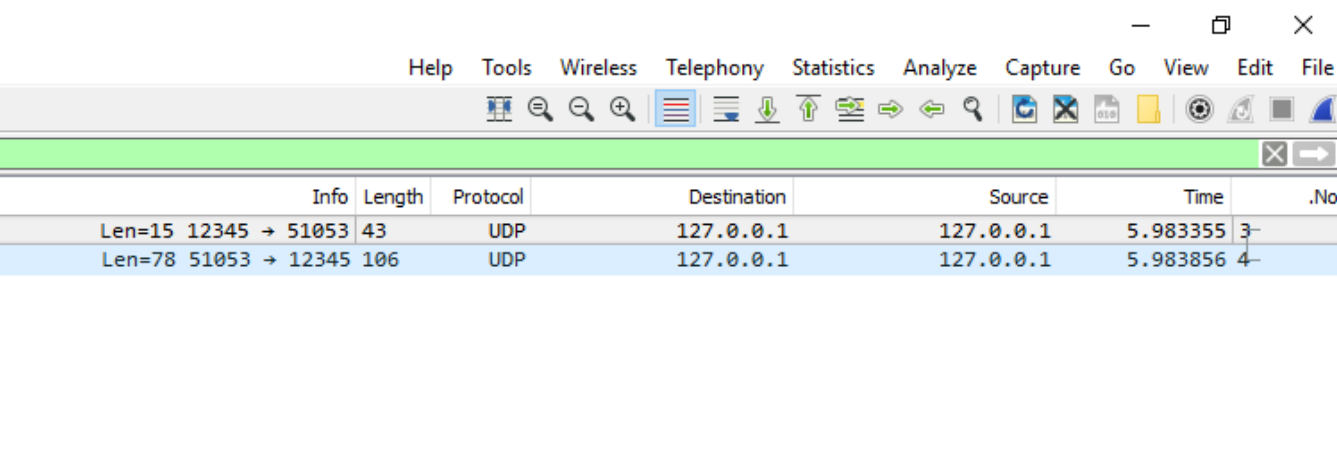
## בצד השרת :

קבלנו הודעה מסוג A מהלקוח (אכן רואים את הקו והפורט שלו)



## כעת נראה את הפקטות ש wireshark תופס:

הקובץ נקרא local\_client



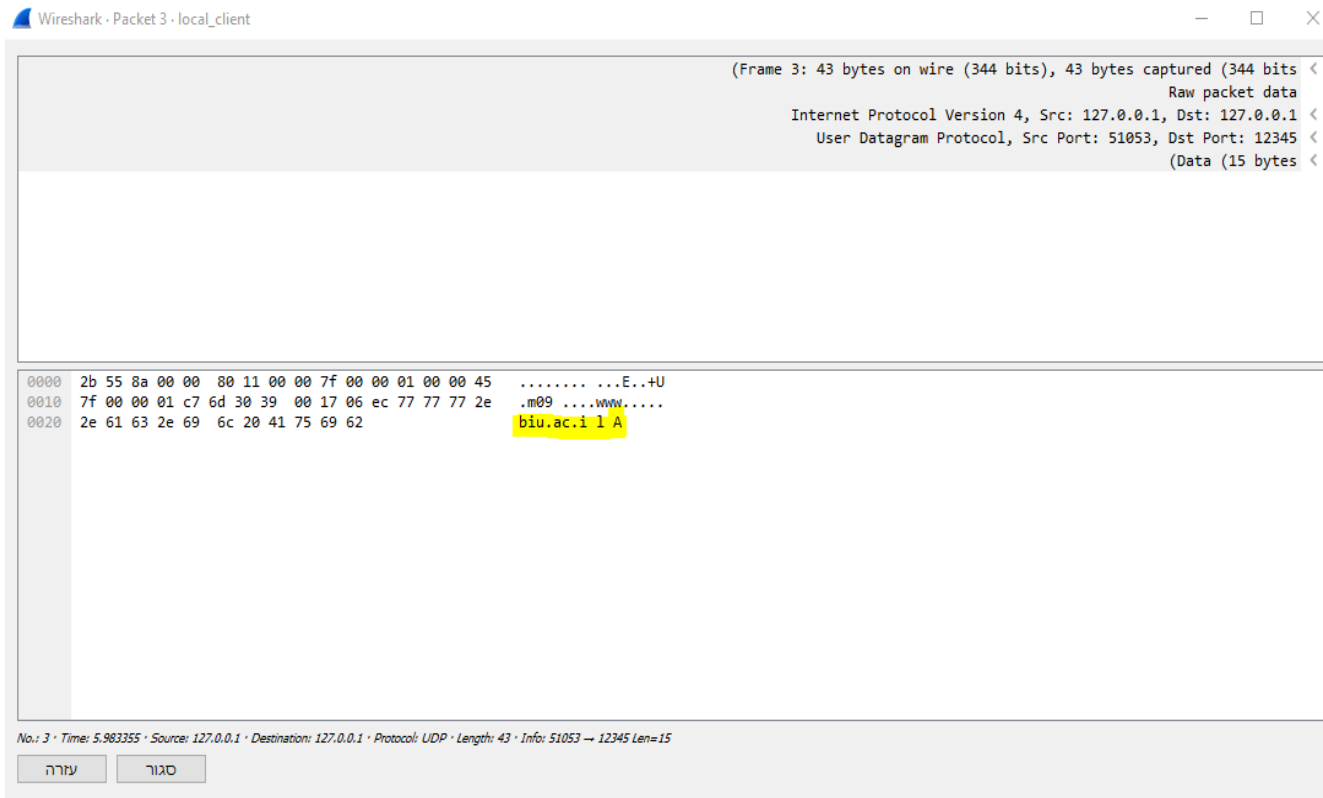
שימו לב שסיננתי לפי קוד מיון שאנחנו מדמים פרוטוקול dns ובמציאות אנחנו עובדים בפרוטוקול udp.

במקרה שלנו הפורט של הלקוח הוא 51053.

הפורט של השרת הוא 12345.

## קבלנו 2 פקטות:

### הראשונה מפורט 51053 ל12345 כלומר מהלקוח לשרת:



קבלנו חבילה של "DNS" וכמובן בנויה לפי מודל השכבות. (נגיד אותם לפי הסדר)

**בשכבת האפליקציה (DATA)** החבילה נקראת Data. שולחים הודעה בפורמט ה"DNS" ששואלים את [www.biu.ac.il](http://www.biu.ac.il) רשומה מסוג A. כלומר מה הקו של [www.biu.ac.il](http://www.biu.ac.il).

**בשכבת התעבורה** (UDP במקרה שלנו). – רואים את פורט המקור והיעד.

במקרה שלנו פורט המקור : 51053, ופורט היעד: 12345.

מכיוון שהיא מדברת על מה קורה אם יש יותר מאפליקציה אחת שרצה על המחשב? במקרה שלנו אפליקציה הכוונה לתהליך ומכיוון שמדובר על אותו מחשב מזהים לפי תהליך ולא לפי IP.

**בשכבת הNetwork (Internet Protocol Version 4)** – לקיחת הודעה ממחשב אחד למחשב אחד.

במקרה שלנו ip המקור וקו היעד בחבילה ייראו אותו הדבר מכיוון שמדובר על אותו מחשב.

## בשכבת הערוץ:

מי המחשב הבא בתור שמקבל את ההודעה.

נראה שם את זמן קבלת החבילה (יהיה הכרחי עוד מעט שנראה בקשר לttl).

**השכבה הפיזית:** מעבירה את החבילה לא רואים אותה זה בעצם איך הביטים עוברים בחומרה.

Wireshark packet capture details for Frame 3. The interface shows the packet structure with various fields and their values.

**Frame 3: 43 bytes on wire (344 bits), 43 bytes captured (344 bits)**

(Encapsulation type: Raw IP (7  
Arrival Time: Dec 3, 2017 23:06:50.896184000  
[Time shift for this packet: 0.000000000 seconds]  
Epoch Time: 1512335210.896184000 seconds  
[Time delta from previous captured frame: 5.982354000 seconds]  
[Time delta from previous displayed frame: 0.000000000 seconds]  
[Time since reference or first frame: 5.983355000 seconds]  
Frame Number: 3  
(Frame Length: 43 bytes (344 bits)  
(Capture Length: 43 bytes (344 bits)  
[Frame is marked: False]  
[Frame is ignored: False]  
[Protocols in frame: raw:ip:udp:data]  
[Coloring Rule Name: UDP]  
[Coloring Rule String: udp]  
Raw packet data  
**Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1**  
Version: 4 = .... 0100  
(Header Length: 20 bytes (5 = 0101 ....  
(Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT <  
Total Length: 43  
(Identification: 0x558a (21898  
Flags: 0x00 <  
Fragment offset: 0  
Time to live: 128  
(Protocol: UDP (17  
[Header checksum: 0x0000 [validation disabled]  
[Header checksum status: Unverified]  
Source: 127.0.0.1  
Destination: 127.0.0.1  
[Source GeoIP: Unknown]  
[Destination GeoIP: Unknown]  
**User Datagram Protocol, Src Port: 51053, Dst Port: 12345**  
Source Port: 51053  
Destination Port: 12345  
Length: 23  
[Checksum: 0x06ec [unverified]  
[Checksum Status: Unverified]  
[Stream index: 0]  
**(Data (15 bytes)**  
Data: 7777772e6269752e616332e696c2041  
[Length: 15]

## השניה מפורט 12345 לפורט 51053 כלומר מהשרת ללקוח

Wireshark · Packet 4 · local\_client

(Frame 4: 106 bytes on wire (848 bits), 106 bytes captured (848 bits) on interface 0  
Raw packet data  
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1  
User Datagram Protocol, Src Port: 12345, Dst Port: 51053  
(Data (78 bytes))

0000	6a 55 8b 00 00 80 11 00 00 7f 00 00 01 00 00 45	.....E..jU
0010	7f 00 00 01 30 39 c7 6d 00 56 ca 4c 54 68 65 20	m.V.LThe.09...
0020	6e 73 77 65 72 20 66 6f 72 20 77 77 77 2e 62 61	answer f or www.b
0030	2e 61 63 2e 69 6c 20 69 73 20 31 2e 32 2e 75 69	.iu.ac.il is 1.2
0040	2e 34 40 77 77 77 2e 62 69 75 2e 61 63 2e 69 33	www. biu.ac.i@3.4
0050	6c 2c 41 2c 31 2e 32 2e 33 2e 34 3a 31 32 33 31	l,A,1.2. 3.4:1231
0060	3a 31 32 33 31 2c 32 30 30 30	00 1231,20:

**בשכבת האפליקציה (DATA)** קבלנו תשובה לשאילתה- כתובת האתר

[www.biu.ac.il](http://www.biu.ac.il) היא 1.2.3.4

**בשכבת התעבורה (UDP)** במקרה שלנו).— רואים את פורט המקור והיעד.

במקרה שלנו פורט המקור : 12345, ופורט היעד: 51053.

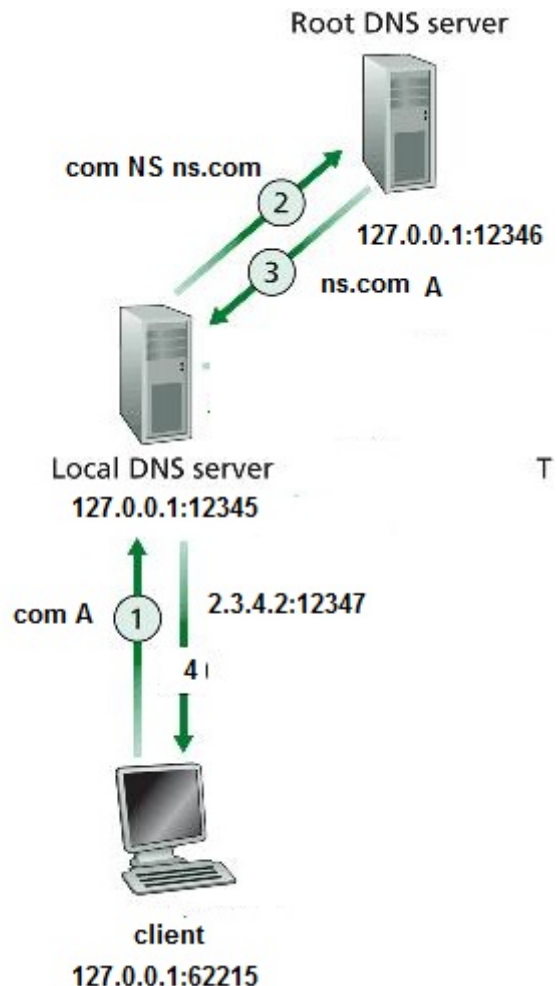
**בשכבת הNetwork (Internet Protocol Version 4)**

במקרה שלנו ip המקור וip היעד בחבילה ייראו אותו הדבר מכיוון שמדובר על אותו מחשב. 127.0.0.1

## לקוח + שרת לוקאלי + שרת אב

במקרה שלנו הפורט של הלקוח 53545 הפורט של השרת הוא 12345.

הפורט של שרת root הוא 12346.



הערה: בשרטוט בטעות שמתי את ns.com במקום לא נכון.

נבקש שאילתת A מהשרת עבור ns.com.

שלחנו בקשת A ואנחנו מצפים לקבל תשובה מהשרת.

Run:	server	server	client
	C:\Python27\python.exe "C:/Users/eliad1998/Documents/2 ניל		
	Message to send: ns.com A		
	Server sent: The answer for ns.com A is 2.3.4.2		

אכן קבלנו תשובה מהשרת נראה איך זה נראה בצד השרת.

כעת נשלחת בקשה לשרת המקומי:

מכיוון שהוא לא מצא ברשומות שלו והוא ריסולבר נעביר את הבקשה לשרת root.

```
Run: server server client
Message: ns.com A from: ('127.0.0.1', 53545)
Message: The answer for com is ns.com from: ('127.0.0.1', 12346)
Message: The answer for ns.com is 2.3.4.2 from: ('127.0.0.1', 12346)
Message: ns.com A from: ('127.0.0.1', 53545)
```

קבלנו הודעה מ127.0.0.1:53545 כלומר מהלקוח

הבקשה הופנתה לשרת root:

השרת אינו מכיל תשובה לכך בקובץ המיפויים שלו לכן יעביר את הבקשה לroot (השרת המקומי שלנו ריסולבר כמובן ואם לא מכיל בקובץ המיפויים יכול להכיל הפניה לשרת אחר)

נבצע בקשת ns עבור com בשביל למצוא את שרת השמות שלו ואז root יחזיר את הip של שרת השמות.

```
Run: server server client
Message: ns.com A from: ('127.0.0.1', 12345)
```

קבלנו בקשת NS כמו שאמרנו מפורט 12345 שהוא השרת הלוקאלי.

בתמונה הקודמת ראינו תשובה מ12346 עבור ns.com כלומר תשובה משרת root אליו.

ואז השרת הלוקאלי מעביר את התשובה ללקוח: כתובת הip של com היא 2.3.4.2

איך כל זה נראה בווירשארק

כמו שעשינו בשרטוט למעלה יהיו סה"כ 4 פקטות.

לקובץ pcap שלנו קוראים client\_local\_root\_ttl.pcap



Info	Length	Protocol	Destination	Source	Time	No.
Len=8 12345 → 53545	36	UDP	127.0.0.1	127.0.0.1	0.000000	1
Len=15 12346 → 12345	43	UDP	127.0.0.1	127.0.0.1	0.005003	2
Len=110 12345 → 12346	138	UDP	127.0.0.1	127.0.0.1	0.005003	3
Len=73 53545 → 12345	101	UDP	127.0.0.1	127.0.0.1	0.005003	4

## הראשונה מפורט 53345 ל12345 כלומר מהלקוח לשרת הלוקאלי

Wireshark · Packet 1 · client\_local\_root\_ttl

(Frame 1: 36 bytes on wire (288 bits), 36 bytes captured (288 bits) on interface 0  
Raw packet data  
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1  
User Datagram Protocol, Src Port: 53545, Dst Port: 12345  
(Data (8 bytes))

```

0000  7c 00 00 80 11 00 00 7f 00 00 01 60 24 00 00 45  .... |$.E
0010  7f 00 00 01 d1 29 30 39 00 10 d3 e3 6e 73 2e 63  ns.c.... 09(....
0020  6f 6d 20 41  om A

```

בשכבת האפליקציה (DATA) שאלנו שאילתת A עבור ns.com.

בשכבת התעבורה (UDP במקרה שלנו). – רואים את פורט המקור והיעד.

במקרה שלנו פורט המקור : 53545, ופורט היעד: 12345.

בשכבת הNetwork (Internet Protocol Version 4)

במקרה שלנו קי המקור וקי היעד בחבילה יראו אותו הדבר מכיוון שמדובר על אותו מחשב. 127.0.0.1

## השנייה מפורט 12345 12346 כלומר מהשרת הלוקאלי לשרת האב

**בשכבת האפליקציה (DATA)** שאלנו שאילתת NS עבור com במטרה לקבל את הקו של שרת השמות (הפננו את הבקשה לשרת הroot)

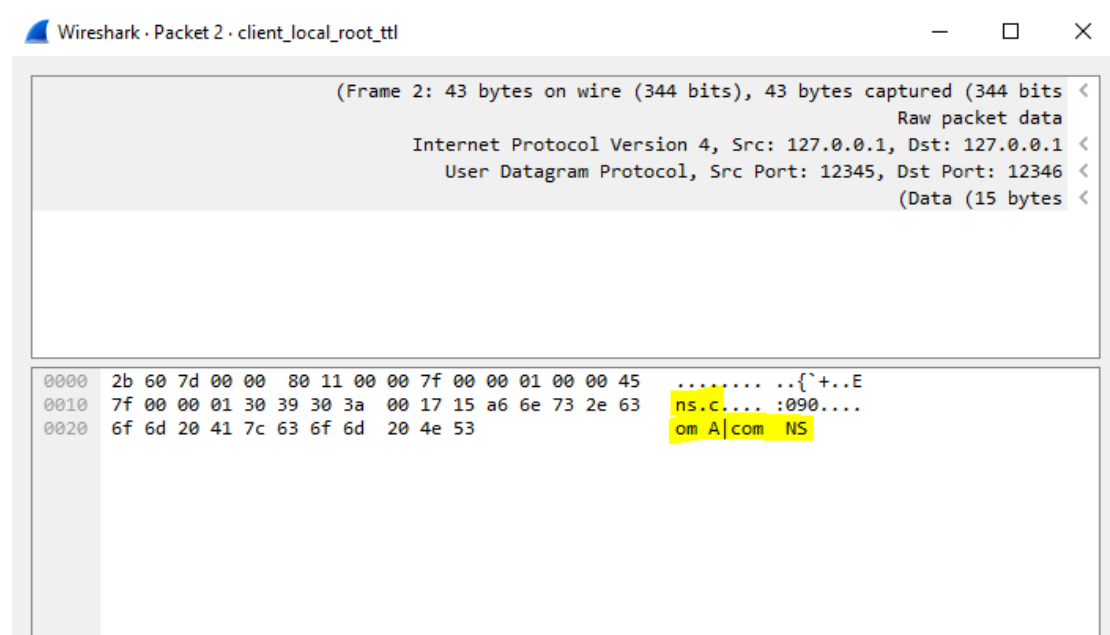
יש את הבקשה של ns.com גם פה עם דגש ששמתי לחפש על com.

**בשכבת התעבורה (UDP במקרה שלנו).** – רואים את פורט המקור והיעד.

במקרה שלנו פורט המקור : 12345, ופורט היעד: 12346.

### בשכבת הNetwork (Internet Protocol Version 4)

במקרה שלנו קו המקור וקו היעד בחבילה יראו אותו הדבר מכיוון שמדובר על אותו מחשב. 127.0.0.1



## השלישית מפורט 12346 12345 כלומר משרת האב לשרת הלוקאלי

**בשכבת האפליקציה (DATA)** שאלנו שאילתת NS עבור com במטרה לקבל את הקו של שרת השמות (הפננו את הבקשה לשרת root)

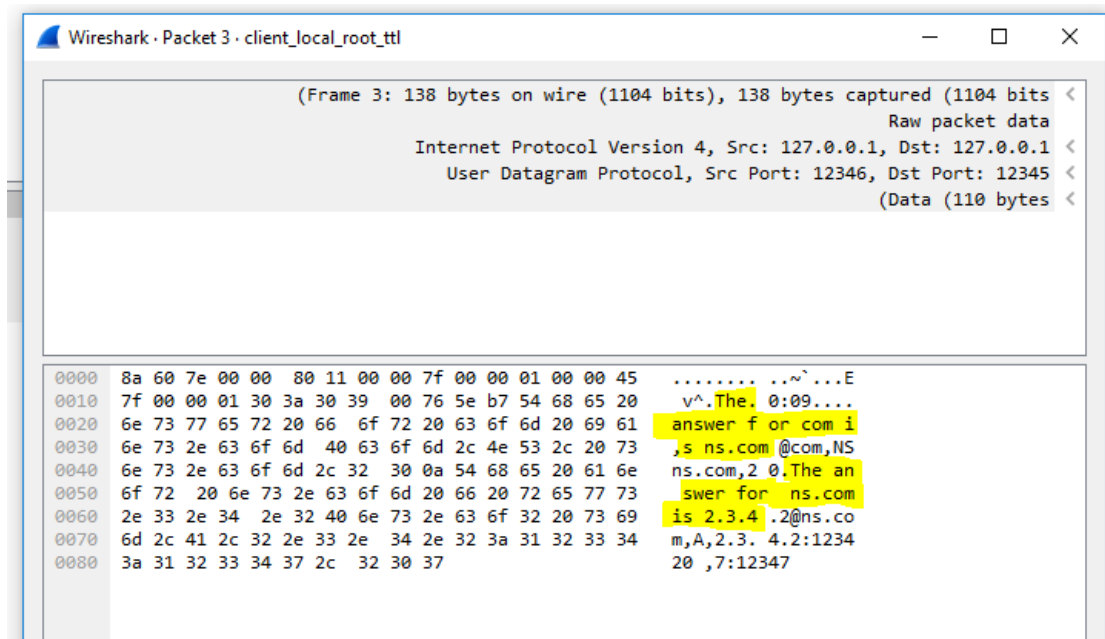
הסבר על המידע המוצג: עשינו שהמידע שישלח בחבילה יכיל בתוכו את כל המידע של המיפוי עבור אותה חבילה בשביל שנשלוק אותו אחר כך ונשמור בקאש. מה שאחרי @ זה מידע על המיפוי בשביל להפוך את זה לאובייקט.

**בשכבת התעבורה (UDP במקרה שלנו).** – רואים את פורט המקור והיעד.

במקרה שלנו פורט המקור : 12346, ופורט היעד: 12345.

### בשכבת הNetwork (Internet Protocol Version 4)

במקרה שלנו ip המקור וקו היעד בחבילה ייראו אותו הדבר מכיוון שמדובר על אותו מחשב. 127.0.0.1



## הרביעית מפורט 53345 12345 כלומר מהשרת הלוקאלי ללקוח

בשכבת האפליקציה (DATA) קבלנו תשובה.

כתובת ה־ip של com היא 1.2.3.4

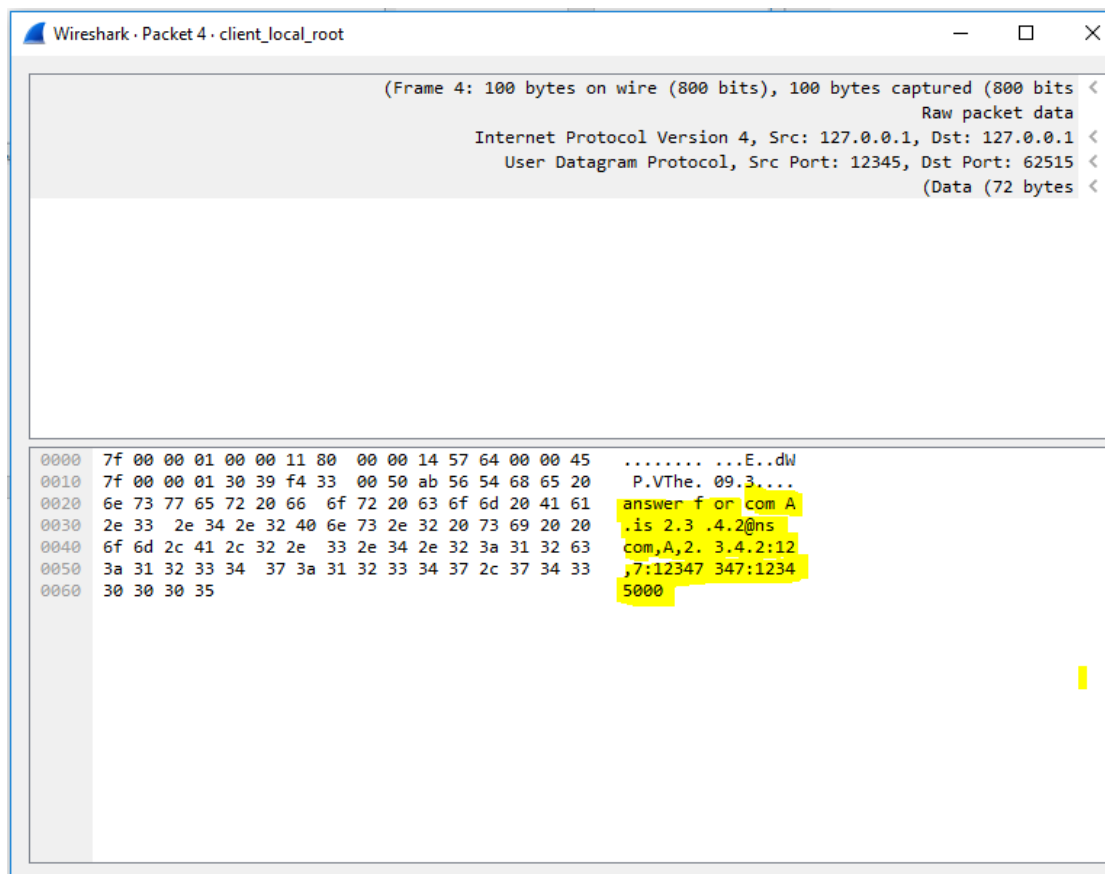
הסבר על המידע המוצג: עשינו שישלח בחבילה יכול בתוכו את כל המידע של המיפוי עבור אותה חבילה בשביל שנשלח אותו אחר כך ונשמור בקאש. מה שאחרי @ זה מידע על המיפוי בשביל להפוך את זה לאובייקט.

**בשכבת התעבורה (UDP במקרה שלנו).** – רואים את פורט המקור והיעד.

במקרה שלנו פורט המקור : 12345, ופורט היעד: 53345.

## בשכבת Network (Internet Protocol Version 4)

במקרה שלנו ip המקור ו־ip היעד בחבילה ייראו אותו הדבר מכיוון שמדובר על אותו מחשב. 127.0.0.1



## איך הtlsl בא פה לידי ביטוי

ראשית שלחנו 3 פעמים בקשה של ns.com A:

```
Run: server server client
C:\Python27\python.exe "C:/Users/eliad1998/Documents/2 תרגילי בית/תרגיל 2
Message to send: ns.com A
Server sent: The answer for ns.com A is 2.3.4.2
Message to send: ns.com A
The answer for ns.com A is 2.3.4.2
Message to send: ns.com A
Server sent: The answer for ns.com A is 2.3.4.2
Message to send:
```

בפעם הראשונה בקשה ראשונה רגילה.

פעם שנייה שימוש בקאש- מכיוון שזה רק root client local אז ישתמש בקאש שבקליינט ולא תהיה חבילה מתאימה.

פעם שלישית אחרי שנגמר הtlsl.

נראה שאפילו לשרת הלוקאלי הגיעו כולה פעמיים בקשות: (כל בלוק בצבע זה בלוק הודעות שמגיע כל פעם של קריאה לשרת עם שרשור לשרת אחר)

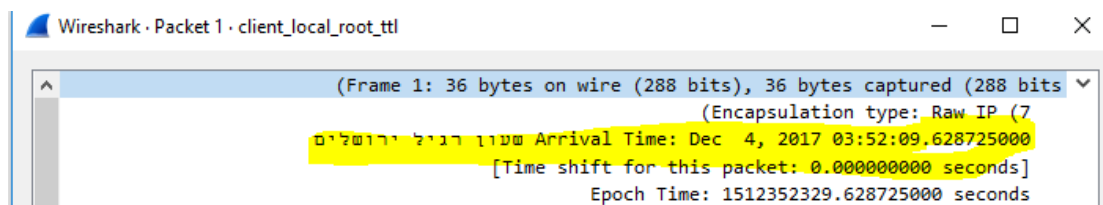
```
Run: server server client
Message: ns.com A from: ('127.0.0.1', 53545)
Message: The answer for com is ns.com from: ('127.0.0.1', 12346)
Message: The answer for ns.com is 2.3.4.2 from: ('127.0.0.1', 12346)
Message: ns.com A from: ('127.0.0.1', 53545)
Message: The answer for com is ns.com from: ('127.0.0.1', 12346)
Message: The answer for ns.com is 2.3.4.2 from: ('127.0.0.1', 12346)
```

## איך נראה בוורשארק:

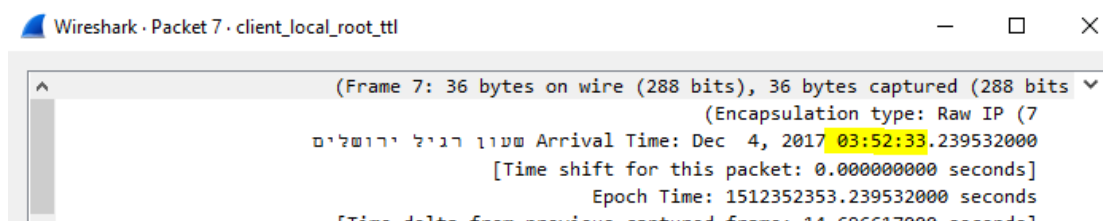
Info	Length	Protocol	Destination	Source	Time	
Len=8 12345 → 53545	36	UDP	127.0.0.1	127.0.0.1	0.000000	1
Len=15 12346 → 12345	43	UDP	127.0.0.1	127.0.0.1	0.005003	2
Len=110 12345 → 12346	138	UDP	127.0.0.1	127.0.0.1	0.005003	3
Len=73 53545 → 12345	101	UDP	127.0.0.1	127.0.0.1	0.005003	4
Len=8 12345 → 53545	36	UDP	127.0.0.1	127.0.0.1	23.610807	7
Len=15 12346 → 12345	43	UDP	127.0.0.1	127.0.0.1	23.610807	8
Len=110 12345 → 12346	138	UDP	127.0.0.1	127.0.0.1	23.610807	9
Len=73 53545 → 12345	101	UDP	127.0.0.1	127.0.0.1	23.611306	10

שוב, פעמיים רק תתבצע שליחה וקבלה כי הפעם השלישית תהיה דרך הקליינט.

נסתכל בשכבת frame של הפקטה הראשונה:



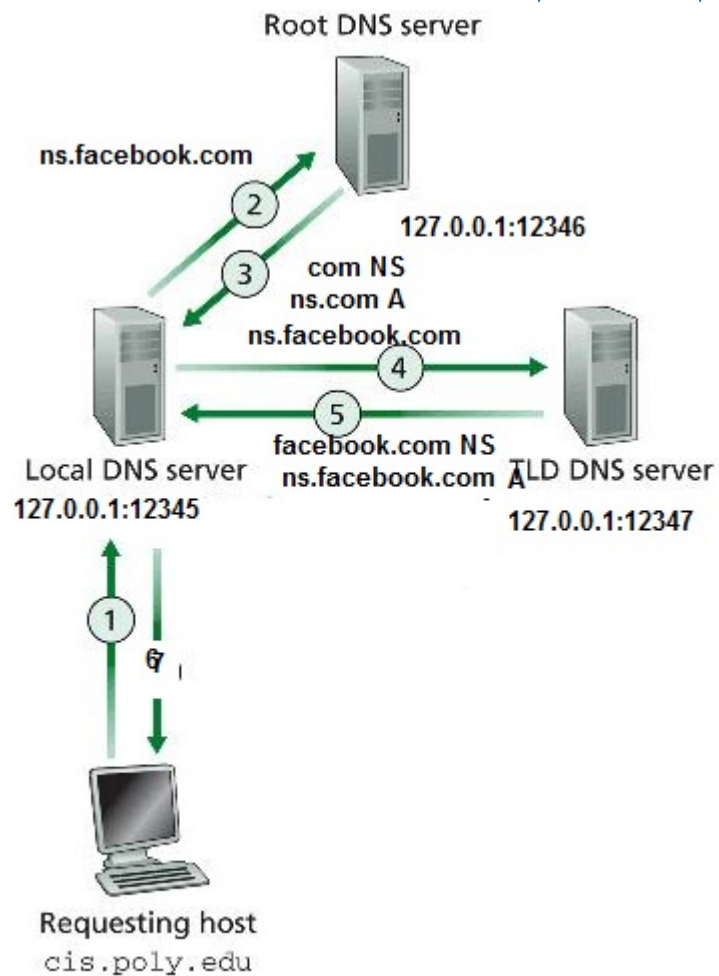
נסתכל בשכבת frame של הפקטה הראשונה של הבקשה של אחרי זה. כלומר, הפקטה החמישית:



נראה כי יש בערך 20 ומשהו שניות בין הפקטות- הגדרתי בקובץ mapping את ttl של ns.com להיות 20 שניות בשביל הרצה זו.

כלומר בזמן הזה עבר ttl ולכן המיפוי נמחק מהקאש והיה צריך לבצע שוב את אותו תהליך.

## לקוח + לוקאלי + שרת אב וגם שרת ייעודי



הלקוח: בפורט 62615

השרת הלוקאלי: בפורט 12345

שרת root: בפורט 12346

שרת tld: בפורט 12347

נראה איך זה נראה בתוכנית:

נבקש את שרת השמות של `facebook.com` ע"י `ns.facebook.com A`. (נניח ואני יודע את שרת השמות)

```
Run: server server server client
C:\Python27\python.exe "C:/Users/eliad1998/Documents/2 ות/תרניל
Message to send: ns.facebook.com A
Server sent: The answer for ns.facebook.com A is 6.3.4.6
Message to send:
```

נקבל תשובה ונראה איך זה הגיע אליה:

```
Run: server server server client
C:\Python27\python.exe "C:/Users/eliad1998/Documents/2 ות/תרניל רשעות/תרניל בית/תרניל
Message: ns.facebook.com A from: ('127.0.0.1', 62615)
Message: The answer for com is ns.com from: ('127.0.0.1', 12346)
Message: The answer for ns.com is 2.3.4.2 from: ('127.0.0.1', 12346)
Message: The answer for facebook.com is ns.facebook.com from: ('127.0.0.1', 12347)
Message: The answer for ns.facebook.com is 6.3.4.6 from: ('127.0.0.1', 12347)
```

מהאיטרטיביות שלחנו אותה בקשה לשרת הלוקאלי.

ב2 שורות למטה יש תשובה עבור חיפוש בשרת השמות של com (לא מצאנו את facebook אז נחפש ב root את com)

```
Run: server server server client
C:\Python27\python.exe "C:/Users/eliad1998/Documents/2 ות/תרניל
Message: ns.facebook.com A from: ('127.0.0.1', 12345)
```

זה שרת root שקיבל את הבקשה האיטרטיבית מ12345- השרת המקומי

אחרי זה 2 שורות למטה בlocal נראה תשובה עבור ns.facebook.com אותה נעביר ללקוח.

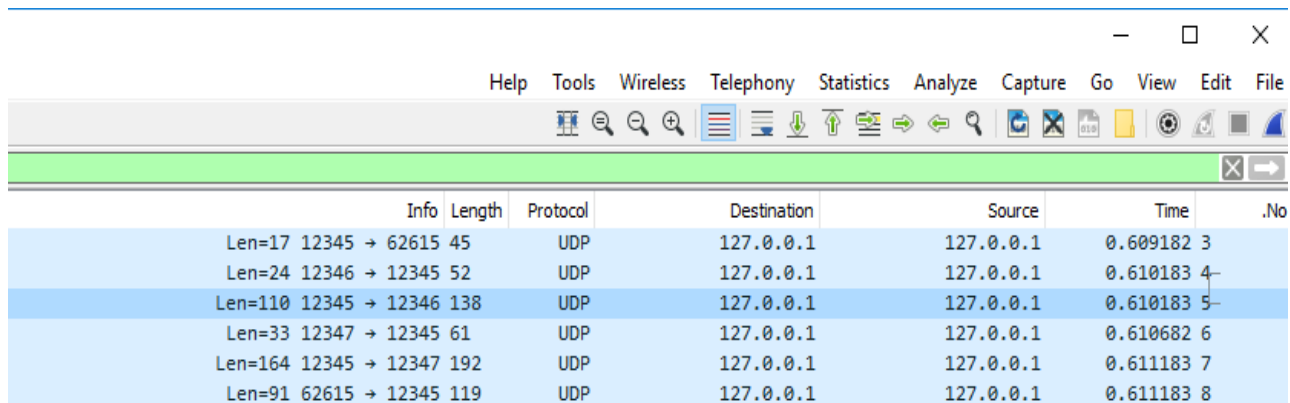
כי אכן חפשנו בldd עבור שרת השמות של facebook ומשם לקו שלו:

```
Run: server server server client
C:\Python27\python.exe "C:/Users/eliad1998/Documents/2 ות/תרניל
Message: ns.facebook.com A from: ('127.0.0.1', 12345)
```



איך זה נראה בוויירשארק:

לקובץ קוראים client\_local\_root\_tld



No.	Time	Source	Destination	Protocol	Length	Info
3	0.609182	127.0.0.1	127.0.0.1	UDP	45	Len=17 12345 → 62615
4	0.610183	127.0.0.1	127.0.0.1	UDP	52	Len=24 12346 → 12345
5	0.610183	127.0.0.1	127.0.0.1	UDP	138	Len=110 12345 → 12346
6	0.610682	127.0.0.1	127.0.0.1	UDP	61	Len=33 12347 → 12345
7	0.611183	127.0.0.1	127.0.0.1	UDP	192	Len=164 12345 → 12347
8	0.611183	127.0.0.1	127.0.0.1	UDP	119	Len=91 62615 → 12345

אכן קבלנו 6 חבילות כדרוש.

שלושת החבילות הראשונות בעמוד הבא.

והשלוש האחרות עוד 2 עמודים.

אפשר למפות למה הפורטים מתאימים לפי המיפוי שרשמתי בעמודים קודמים כדי לראות שאני באמת צודק.

### חבילה 1:

מהלקוח לשרת המקומי – שואלת אותו מה הִק של ns.facebook.com.

### חבילה 2:

מהשרת המקומי לשרת הִroot: השרת המקומי לא יודע אז מתחיל באיטרטיביות ושואל את root אותה שאלה.

### חבילה 3:

משרת הִroot לשרת המקומי: מחזיר לו את הִק של שרת השמות של .com.

### חבילה 4:

מהשרת המקומי לִtld.

כעת השרת המקומי יודע ישר לפנות לִtld (במקרה שלנו שרת השמות של .com) לפי השאילתה הקודמת. נשאל את הִtld את הבקשה האיטרטיבית.

### חבילה 5:

מהשרת הtld לשרת הlוקאלי: מחזיר לו את התשובה עבור שרת השמות של  
.facebook.com

חבילה 6:

מהשרת הlוקאלי למשתמש: מחזיר לו את התשובה לשאלתה שביקש.

Wireshark - Packet 4 - local\_server\_root\_tld

(Frame 4: 52 bytes on wire (416 bits), 52 bytes captured (416 bits) <  
Raw packet data  
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1 <  
User Datagram Protocol, Src Port: 12345, Dst Port: 12346 <  
(Data (24 bytes) <

0000 7f 00 00 01 00 00 11 80 00 00 53 61 34 00 00 45 ..... ..E...aS  
0010 7f 00 00 01 30 39 30 3a 00 20 32 0c 6e 73 2e 66 ns.f.2. :090...  
0020 6f 6f 6b 2e 63 6f 6d 20 41 7c 63 6f 62 65 63 61 acebook. com A) co  
0030 6d 20 4e 53 m NS

No.: 4 • Time: 0.610183 • Source: 127.0.0.1 • Destination: 127.0.0.1 • Protocol: UDP • Length: 52 • Info: 12345 → 12346 Len=24

עזרה סגור

Wireshark - Packet 3 - local\_server\_root\_tld

(Frame 3: 45 bytes on wire (360 bits), 45 bytes captured (360 bits) <  
Raw packet data  
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1 <  
User Datagram Protocol, Src Port: 62615, Dst Port: 12345 <  
(Data (17 bytes) <

0000 2d 61 52 00 00 80 11 00 00 7f 00 00 01 00 00 45 ..... ..E...aR  
0010 7f 00 00 01 f4 97 30 39 00 19 8d 1b 6e 73 2e 66 ns.f.... 09.....  
0020 6f 6f 6b 2e 63 6f 6d 20 41 62 65 63 61 acebook. com A

No.: 3 • Time: 0.609182 • Source: 127.0.0.1 • Destination: 127.0.0.1 • Protocol: UDP • Length: 45 • Info: 62615 → 12345 Len=17

עזרה סגור

Wireshark - Packet 5 - local\_server\_root\_tld

(Frame 5: 138 bytes on wire (1104 bits), 138 bytes captured (1104 bits) <  
Raw packet data  
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1 <  
User Datagram Protocol, Src Port: 12346, Dst Port: 12345 <  
(Data (110 bytes) <

0000 8a 61 54 00 00 80 11 00 00 7f 00 00 01 00 00 45 ..... ..E...aT  
0010 7f 00 00 01 30 3a 30 39 00 76 5e b7 54 68 65 20 v^The. 0:09...  
0020 6e 73 77 65 72 20 66 6f 72 20 63 6f 6d 20 69 61 answer f or com i  
0030 6e 73 2e 63 6f 6d 40 63 6f 6d 2c 4e 53 2c 20 73 ,s ns.com@com,NS  
0040 6e 73 2e 63 6f 6d 2c 32 30 0a 54 68 65 20 61 6e ns.com,2 0.The an  
0050 6f 72 20 6e 73 2e 63 6f 6d 20 66 20 72 65 77 73 swer for ns.com  
0060 2e 33 2e 34 2e 32 40 6e 73 2e 63 6f 32 20 73 69 is 2.3.4.2@ns.co  
0070 6d 2c 41 2c 32 2e 33 2e 34 2e 32 3a 31 32 33 34 m,A,2.3. 4.2:1234  
0080 3a 31 32 33 34 37 2c 32 30 37 20 ,7:12347

No.: 5 • Time: 0.610183 • Source: 127.0.0.1 • Destination: 127.0.0.1 • Protocol: UDP • Length: 138 • Info: 12346 → 12345 Len=110

Wireshark - Packet 7 - local\_server\_root\_tld

(Frame 7: 192 bytes on wire (1536 bits), 192 bytes captured (1536 bits) on interface 0  
Raw packet data  
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1  
User Datagram Protocol, Src Port: 12347, Dst Port: 12345  
(Data (164 bytes))

Handwritten '5' in the packet details pane.

0000 c0 61 56 00 00 80 11 00 00 7f 00 00 01 00 00 45 .....E...aV  
0010 7f 00 00 01 30 3b 30 39 00 ac d6 3b 54 68 65 20 The;...09;0....  
0020 6e 73 77 65 72 20 66 6f 72 20 66 61 63 65 62 61 answer f or faceb  
0030 6f 6f 6b 2e 63 6f 6d 20 69 73 20 6e 73 2e 66 61 ook.com is ns.fa  
0040 6f 6f 6b 2e 63 6f 6d 40 66 61 63 65 62 62 65 63 cebook.c om@faceb  
0050 6f 6f 6b 2e 63 6f 6d 2c 4e 53 2c 6e 73 2e 66 61 ook.com, NS,ns.fa  
0060 6f 6f 6b 2e 63 6f 6d 2c 32 30 0a 54 68 62 65 63 cebook.c om,20.Th  
0070 6e 73 77 65 72 20 66 6f 72 20 6e 73 2e 61 20 65 e answer for ns  
0080 6f 6f 6b 2e 63 6f 6d 20 69 73 20 62 65 63 61 66 facebook .com is  
0090 2e 33 2e 34 2e 36 40 6e 73 2e 66 61 63 65 62 36 ns.faceb @6.3.4.6  
00a0 6f 6f 6b 2e 63 6f 6d 2c 41 2c 36 2e 33 2e 34 2e .ook.com, A,6.3.4  
00b0 3a 31 32 33 34 38 3a 31 32 33 34 2c 32 30 36 12348,20 :6:12348

No.: 7 • Time: 0.611183 • Source: 127.0.0.1 • Destination: 127.0.0.1 • Protocol: UDP • Length: 61 • Info: 12345 → 12347 Len=33

עזרה סגור

Wireshark - Packet 6 - local\_server\_root\_tld

(Frame 6: 61 bytes on wire (488 bits), 61 bytes captured (488 bits) on interface 0  
Raw packet data  
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1  
User Datagram Protocol, Src Port: 12345, Dst Port: 12347  
(Data (33 bytes))

Handwritten '4' in the packet details pane.

0000 3d 61 55 00 00 80 11 00 00 7f 00 00 01 00 00 45 .....E...=aU  
0010 7f 00 00 01 30 39 30 3b 00 29 a4 1b 6e 73 2e 66 ns.f...;090....  
0020 6f 6f 6b 2e 63 6f 6d 20 41 7c 66 61 62 65 63 61 acebook. com A|fa  
0030 6f 6f 6b 2e 63 6f 6d 20 4e 53 62 65 63 63 cebook.c om NS

Wireshark - Packet 8 - local\_server\_root\_tld

(Frame 8: 119 bytes on wire (952 bits), 119 bytes captured (952 bits) on interface 0  
Raw packet data  
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1  
User Datagram Protocol, Src Port: 12345, Dst Port: 62615  
(Data (91 bytes))

Handwritten '6' in the packet details pane.

0000 7f 00 00 01 00 00 11 80 00 00 57 61 77 00 00 45 .....E...waW  
0010 7f 00 00 01 30 39 f4 97 00 63 28 b8 54 68 65 20 c(.The. .09....  
0020 6e 73 77 65 72 20 66 6f 72 20 6e 73 2e 66 61 61 answer f or ns.fa  
0030 6f 6f 6b 2e 63 6f 6d 20 41 20 20 69 73 62 65 63 cebook.c om A is  
0040 2e 33 2e 34 2e 36 40 6e 73 2e 66 61 63 65 36 20 ns.face@ 6.3.4.6  
0050 6f 6f 6b 2e 63 6f 6d 2c 41 2c 36 2e 33 2e 34 62 book.com ,A,6.3.4  
0060 2e 36 3a 31 32 33 34 38 3a 31 32 33 34 38 3a 31 12348:1: 6:12348.  
0070 2c 32 30 38 34 33 32 2348,20

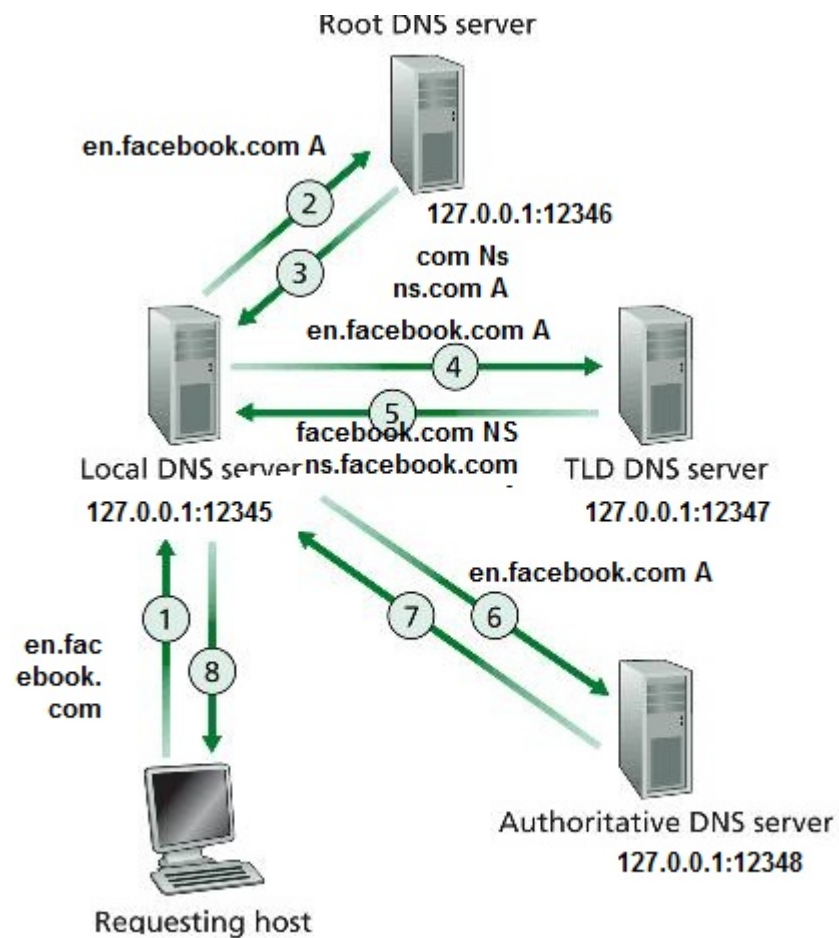
rootn לשרת המקומי: מחזיר לו את ה<sup>קו</sup> של שרת

4:

: המקומי tld.

שרת המקומי יודע ישר לפנות ל<sup>tld</sup> (במקרה שלנו ש  
פי השאילתה הקודמת. נשאל את ה<sup>tld</sup> את הבקש

לקוח + שרת לוקאלי + שרת אב + שרת ייעודי + שרת ייעודי  
תחתיו בהיררכיה



הלקוח: בפורט 62498

השרת הלוקאלי: בפורט 12345

שרת הroot: בפורט 12346

שרת הtld: בפורט 12347

שרת הauth: בפורט 12348

נראה איך זה נראה בתוכנית:

נבקש את A en.facebook.com.

```
Run: server server server server client
C:\Python27\python.exe "C:/Users/eliad1998/Documents/2 טח/רשתות/תרגילי בית/תרגיל 2
Message to send: en.facebook.com A
Server sent: The answer for en.facebook.com is 3.2.5.4
Message to send:
```

זה השרת הלוקאלי אליו חוזרות התשובות משרת ה root עבור com, משרת  
ה tld עבור facebook.com ומשרת האuth עבור הכתובת המלאה  
en.facebook.com.

```
Run: server server server server client
C:\Python27\python.exe "C:/Users/eliad1998/Documents/2 אוניברסיטה/רשתות/תרגילי בית/תרגיל 2/dnsServer/serve
Message: en.facebook.com A from: ('127.0.0.1', 62498)
Message: The answer for com is ns.com from: ('127.0.0.1', 12346)
Message: The answer for ns.com is 2.3.4.2 from: ('127.0.0.1', 12346)
Message: The answer for facebook.com is ns.facebook.com from: ('127.0.0.1', 12347)
Message: The answer for ns.facebook.com is 6.3.4.6 from: ('127.0.0.1', 12347)
Message: The answer for en.facebook.com is 3.2.5.4 from: ('127.0.0.1', 12348)
```

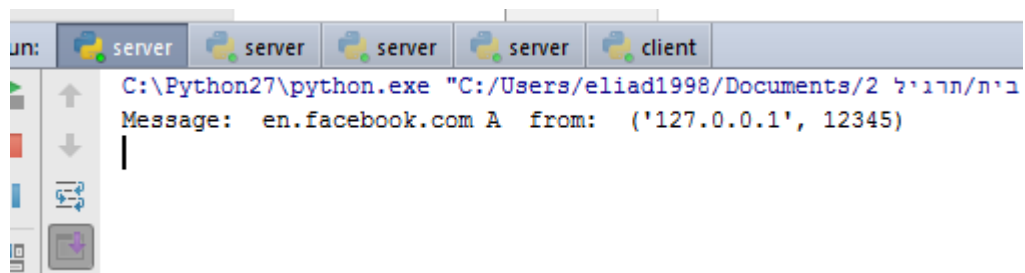
הפנייה משרת הלוקאלי לשרת הרוט.

```
Run: server server server server client
C:\Python27\python.exe "C:/Users/eliad1998/Documents/2 תרגיל /
Message: en.facebook.com A from: ('127.0.0.1', 12345)
```

הפנייה משרת הלוקאלי לשרת הtld.

```
Run: server server server server client
C:\Python27\python.exe "C:/Users/eliad1998/Documents/2 ית/תרגיל /
Message: en.facebook.com A from: ('127.0.0.1', 12345)
```

הפנייה מהשרת הלוקאלי לשרת האuth.



נראה איך זה נראה בwireshark:

קובץ pcap שלנו נקרא local\_server\_root\_tld.

No.	Time	Source	Destination	Protocol	Length	Info
3	0.851214	127.0.0.1	127.0.0.1	UDP	45	Len=17 12345 → 62498
4	0.851713	127.0.0.1	127.0.0.1	UDP	52	Len=24 12346 → 12345
5	0.852214	127.0.0.1	127.0.0.1	UDP	138	Len=110 12345 → 12346
6	0.852714	127.0.0.1	127.0.0.1	UDP	61	Len=33 12347 → 12345
7	0.853214	127.0.0.1	127.0.0.1	UDP	192	Len=164 12345 → 12347
8	0.853715	127.0.0.1	127.0.0.1	UDP	63	Len=35 12348 → 12345
9	0.854215	127.0.0.1	127.0.0.1	UDP	112	Len=84 12345 → 12348
10	0.854215	127.0.0.1	127.0.0.1	UDP	118	Len=90 62498 → 12345

יש לנו 8 פקטות כצפוי.

**חבילה 1:** מלקוח לשרת הלוקאלי שואלת אותו תביא לי את כתובת הip של en.facebook.com.

**חבילה 2:** מהשרת הלוקאלי לשרת root: שואלת אותו דבר בצורה איטרטיבית.

**חבילה 3:** משרת root לשרת הלוקאלי: שרת root מביא לו את כתובת הip של שרת השמות של com.

**חבילה 4:** משרת הלוקאלי לשרת tld: שואל אותו בצורה איטרטיבית את אותה בקשה.

**חבילה 5:** משרת הtld לשרת הלוקאלי: מביא לו את כתובת הקו של שרת השמות של facebook.com.

**חבילה 6:** מהשרת הלוקאלי לשרת הauth: שואל אותו אותה בקשה.

**חבילה 7:** הפעם לשרת הauth בקובץ המיפויים שלו יש את התשובה לבקשה שלנו והוא מחזיר אותה.

**חבילה 8:** מהשרת הלוקאלי ללקוח: מביא לו את מה שביקש.

The image displays four Wireshark packet capture windows, each showing a different packet in a sequence. Each window has a title bar indicating the packet number and the interface 'client\_local\_root\_tld\_auth'.

- Packet 3:** Shows a query from the client to the local server. The 'Data' field (length 17) contains the hex representation of the query. The packet list shows the hex data and the ASCII representation: 'en.f...acebook.com A'. A handwritten '1' is next to the packet.
- Packet 4:** Shows a response from the local server to the client. The 'Data' field (length 24) contains the hex representation of the response. The packet list shows the hex data and the ASCII representation: 'en.f...acebook.com A'. A handwritten '2' is next to the packet.
- Packet 5:** Shows a query from the client to the local server. The 'Data' field (length 110) contains the hex representation of the query. The packet list shows the hex data and the ASCII representation: 'en.f...acebook.com A'. A handwritten '3' is next to the packet.
- Packet 6:** Shows a response from the local server to the client. The 'Data' field (length 33) contains the hex representation of the response. The packet list shows the hex data and the ASCII representation: 'en.f...acebook.com A'. A handwritten '4' is next to the packet.

