בס"ד

תקשורת ומחשוב מטלת סיום

מגישות:

ליאור נגר ת"ז: 209399294

רז אלבז ת"ז: 207276775

מבוא:

בס"ד

חלקים א',ב':

<u>הסבר כללי:</u>

המטלה מחולקת לשלושה חלקים

חלק א – בניית מערכת מסרים מיידים פרימיטיבית מבוססת על תקשורת

חלק ב – הוספה למערכת שבנינו שכבה חדשה להעברת קבצים מעל UDP

חלק ג – שאלות

לצורך חלק א ו-ב השתמשנו במחלקות הבאות:

1. $server$ - מאזין לחיבורים חדשים
2. $client$ – מתחבר אל השרת
3. new_thread_user – מחלקה אשר נקראת ע"י ה server ויוצרת Thread חדש עבור כל חיבור חדש עם client
4. gui - מחלקה של ממשק גרפי לתצוגת הצ'אט שיצרנו

<u>הרצה:</u>

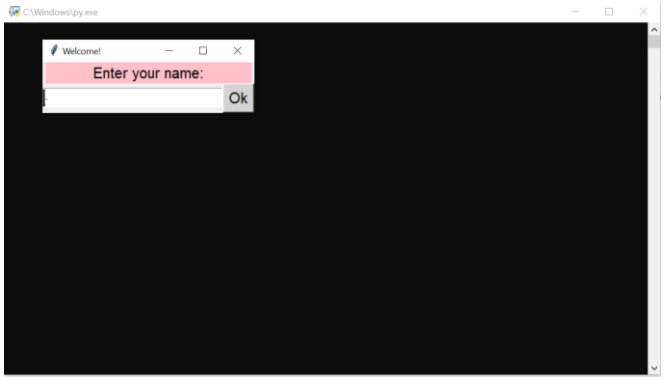
התבקשנו לעמוד במשימות הבאות:
הלקוח (client) יכול לבצע את הפעולות הבאות:

1)להתחבר לשרת.
2)להתנתק מהשרת.
3)לשלוח הודעה ללקוח אחר.
4)לשלוח הודעה לכל הלקוחות המחוברים לשרת כעת.
5)לקבל את שמות הלקוחות המחוברים לשרת.
6)לקבל רשימת קבצים הקיימים בשרת.
7)לשלוח בקשה להורדת קובץ מהשרת.
8)להוריד קובץ מתוך השרת.
כמו כן, במידה והצטרף לקוח חדש או התנתק לקוח קיים יש לשלוח הודעה מתאימה לכל המשתתפים.

במהלך ההסבר על ההרצה נדגיש היכן הן התבצע והוצגו באמצעות ה-GUI(אותן פעולות התרחשו במסכי ה-CLIENENTS וה-SERVER - ניתן לראות בעת ההפעלה של התוכנית)

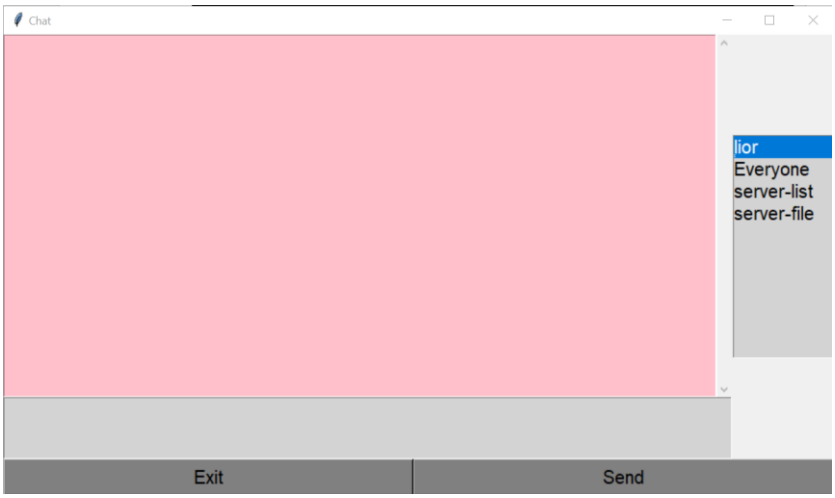
דבר ראשון אנו צריכים להריץ את ה server לאחר מכן נריץ את קובץ ה- client בכדי שהוא יוכל להתחבר אליו. ניתן לפתוח מספר clients שיוכלו לדבר יחד בצ'אט כפי שהתבקשנו במטלה.

ברגע שנריץ את קובץ ה client יפתח המסך הבא:



במסך זה יצטרך הלקוח להזין את שם המשתמש, שם זה ייצג אותו בצ'אט לאחר מכן ברשימת המשתתפים שתופיע בצידו הימני של המסך, וילחץ על OK או enter לבחירתו, בכדי לעבור אל הצ'אט.

לאחר לחיצה על כפתור הOK נקבל את המסך הבא:



כאן התבצעה פעולה מספר 1, הצלחנו להתחבר לשרת ולהיכנס לצ'אט.

לאחר שנחבר קליינט נוסף יפתח חלון צ'אט נוסף עבור הקליינט, והחלון שלנו יתעדכן ויראה כך:

בחיבור של המשתמשת הראשונה 'lior' אין צורך בהודעה לכל המשתתפים היות והיא המשתתפת היחידה. בעת החיבור של המשתמשת השנייה 'raz' , מכיוון שהתחבר משתמש נוסף, נוספה ההודעה אותה התבקשנו לשלוח לכל המשתתפים והיא שהמשתמשת 'raz' התחברה.

כאן התבצעה פעולה מספר 5, כל משתמש יוכל לקבל תמיד את שמות הלקוחות המחוברים כעת בכך שיופיעו מול עיניו ויידע "אונליין" לאילו משתתפים יוכל לשלוח הודעה.

כעת נסביר על החלון:
מצד ימין ניתן לראות את רשימת המחוברים, תמיד יופיעו שמות המחוברים, האופציה $everyone$ ושתי אופציות תקשורת עם השרת:

$server - list$ - בקשת רשימת הקבצים שנמצאים בשרת וזמינים להורדה.

$server - file$- בקשת הורדת קובץ מהשרת אל הלקוח.

בשביל לשלוח הודעות אל מישהו מהרשימה צריך ללחוץ על שמו בעזרת העכבר או בשביל לשלוח הודעה אל כלל המשתתפים נלחץ על השם $Everyone$, לאחר מכן נכתוב את ההודעה בתיבת ההקלדה , ואז נלחץ על $send$ או דרך המקלדת על $enter$. לאחר שליחת ההודעה נראה את ההודעה ששלחנו על מסך ההודעות:

והמסך של הלקוח שקיבל את ההודעה יראה כך:

כאן התבצעה פעולה מספר 3, שהיא לשלוח הודעה ללקוח אחר. המשתמשת 'lior' שלחה הודעה למשתמשת 'raz'.

כעת נתבונן בשליחת הודעה ל$Everyone$, ניתן לראות שבפעולה זו לחוץ בחלון המשתמשים האפשרות הנ"ל, ובכך כל המשתמשים מקבלים את ההודעה מ 'raz' .

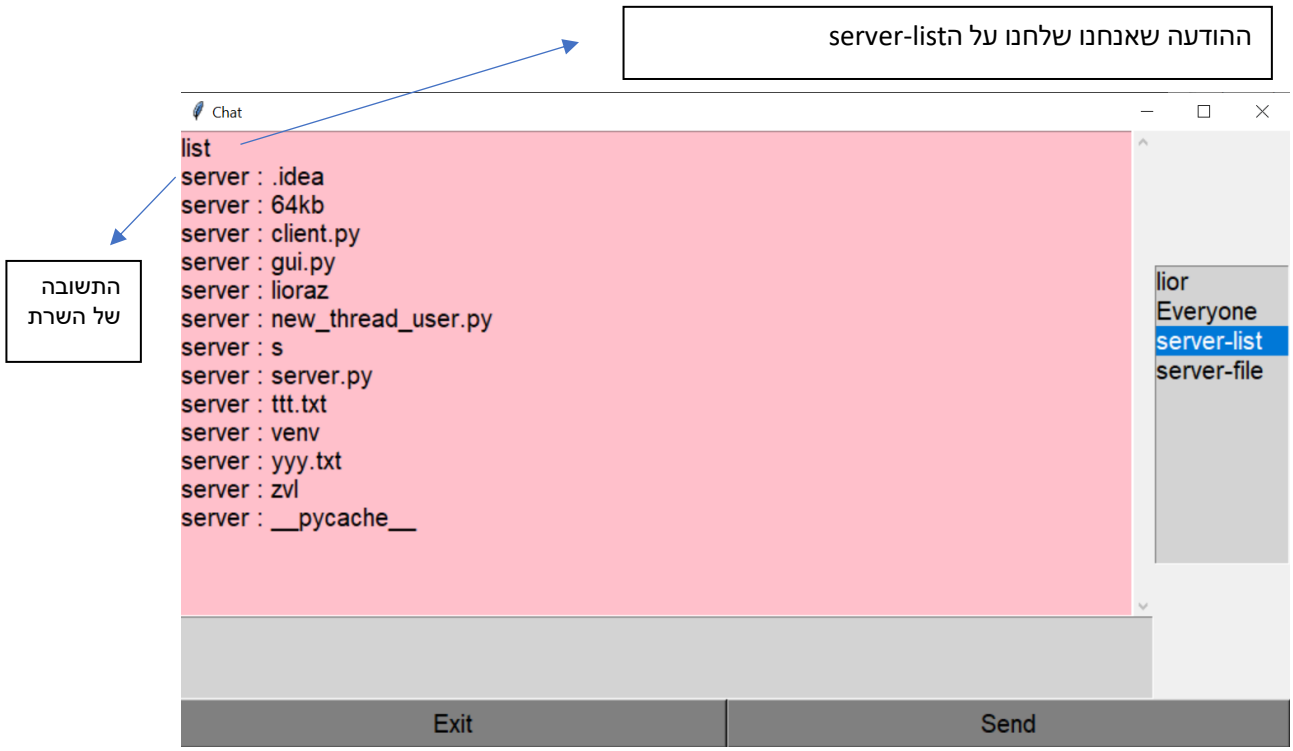
כאן התבצעה פעולה מספר 4, שהיא לשלוח הודעה לכל הלקוחות האחרים המחוברים לשרת כעת.

כעת נציג את הפעולות שמתרחשות כאשר לקוח בוחר להתנתק מהצ'אט (מהשרת) על ידי לחיצת X בחלון או על $Exit$ ברצונו.

כאן התבצעה פעולה מספר 2, שהיא להתנתק מהצ'אט ובכך להתנתק מהשרת. בנוסף, נשלחה ההודעה אותה התבקשנו לשלוח לכל המשתתפים והיא שהמשתמשת 'lior' התנתקה.

נסביר כעת על האפשרות לבקשת את רשימת הקבצים מהשרת:

באשר הקלינט מחובר לשרת ורוצה לבקש ממנו את רשימת הקבצים, הוא צריך לשלוח ל $server - list$ הודעה כלשהי, בצורה חופשית (בכל פעם שירצה להוריד קובץ מהשרת). לאחר מכן הלקוח יקבל את רשימת שמות הקבצים מהשרת ונוכל לראות את הרשימה במסך הצ'אט :



ניתן לראות שקיבלנו הרשימה של הקבצים.

בחלק זה ענינו על פעולה מספר 6, קיבלנו מהשרת את רשימת הקבצים הקיימים בו.

כעת כדי להוריד קובץ מהשרת נשלח הודעה אל $server - file$ ובהודעה נרשום את שם הטקסט שאותו אנו רוצים להוריד.

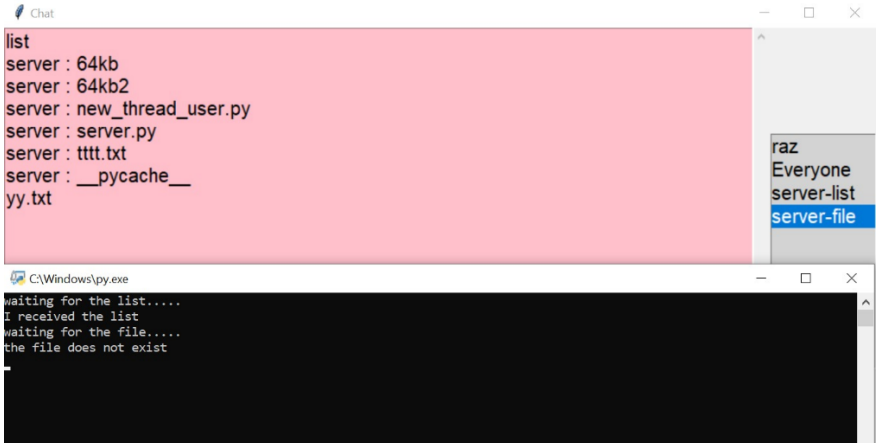
תמונה לפני לחיצה על $send$:

ולאחר שנלחץ $send$ נקבל במסך של הלקוח כי קיבלנו את ההודעה:

בחלק זה ענינו על פעולה מספר 7+8, הלקוח ביקש להוריד את הקובץ וקיבל אותו.

בנוסף, השרת יכול לשלוח את אותו הקובץ במקביל למספר לקוחות וגם לשלוח קבצים שונים ללקוחות במקביל.

כמו כן, במידה והקובץ לא קיים, הלקוח לא יקבל אותו, ניתן לראות זאת בצילום המסך הבא:

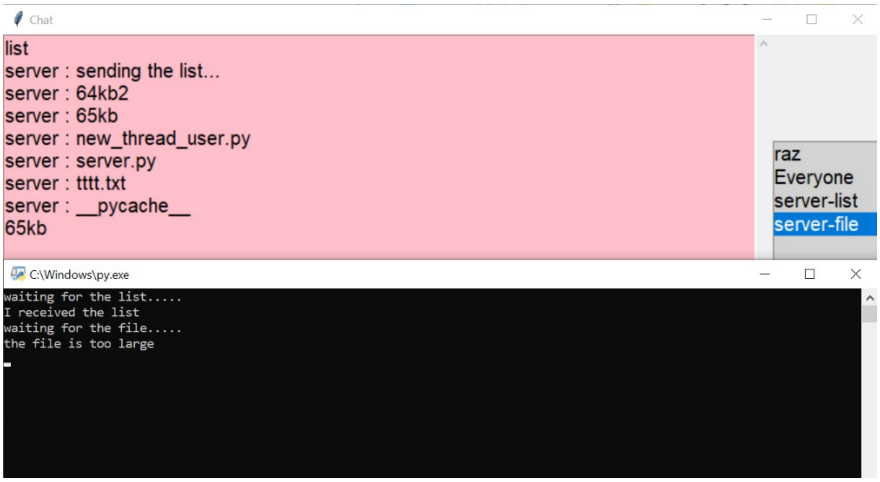


היות והתבקשנו במטלה לשלוח קובץ עד kb64, טיפלנו בנוסף בבעיה שעלולה להיווצר כאשר קובץ גדול מידי שהשרת לא יוכל להעביר:

הקובץ :

65kb	02/03/2022 13:31	קובץ	65 KB

צילומי המסך:



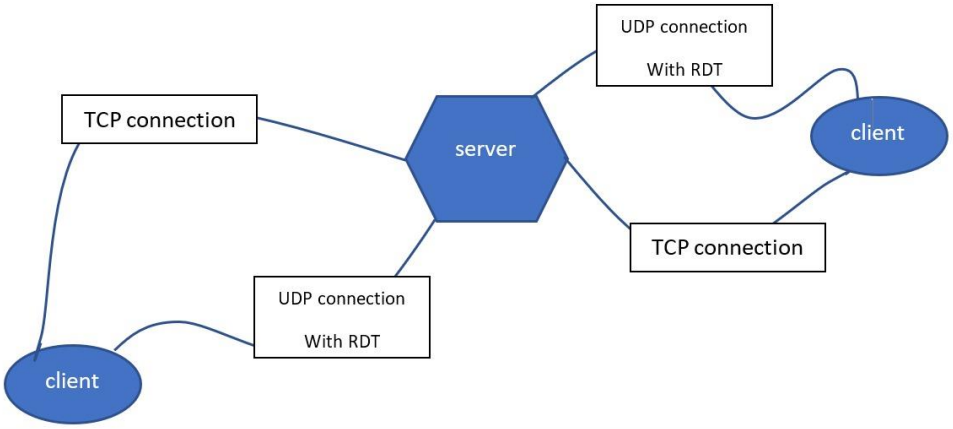
נציין שבעת ההפעלת התוכנית גם מסך ה $server$ וגם מסכי ה$clients$ מקבלים התראות על כל פעולה אשר מתרחשת בצ'אט.

שאלות חלק ב':

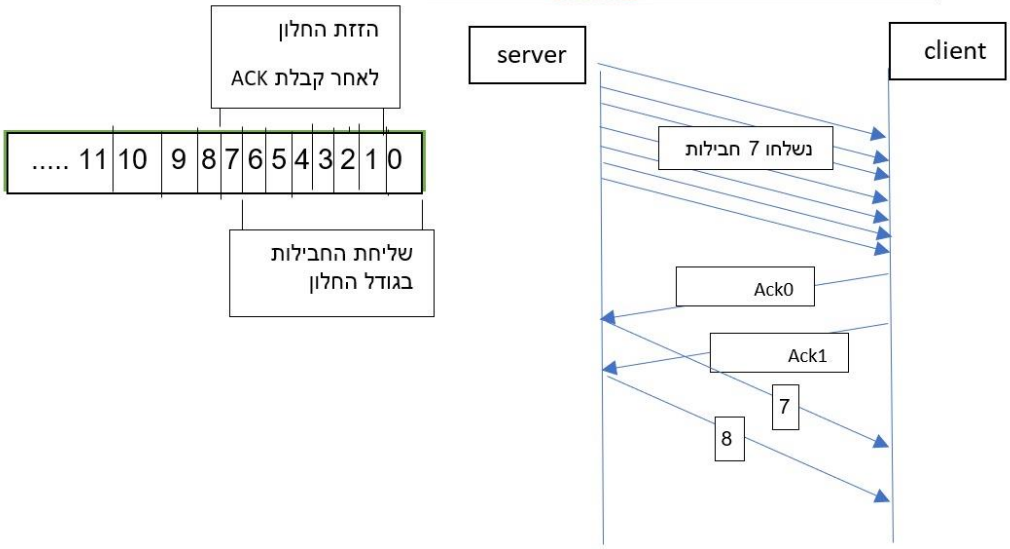
חלק ב'– **על בסיס תיאור המערכת:**
להוסיף לאותה מערכת שכבה חדשה (קוד נוסף) להעברת קבצים מעל UDP אשר נקרא לה FAST reliable UDP
ענו על השאלות הבאות:

- ציירו דיאגרמת מצבים בהם המערכת עובדת
- כיצד המערכת מתגברת על איבוד חבילות
- כיצד המערכת מתגברת על בעיות latency

דיאגרמת מצבים בהן המערכת עובדת:



העברת הקבצים במערכת לפי פרטוקול GO BACK N



כיצד המערכת מתגברת על איבוד חבילות:

למימוש חלק ב השתמשנו בפרוטוקול $GO\ BACK\ N$.

פרוטוקול זה הוא פרוטוקול חלון הזזה אשר משיג ניצול של רוחב הפס של הרשת.

בפרוטוקול זה השולח שולט בקצב זרימת החבילות. לשולח יש רצף של חבילות לשלוח, הוא מגדיר את גודל החלון. (אנו הגדרנו את גודל החלון בגודל 7), והחבילות משודרות בזו אחר זו לפי גודל החלון.

הצד המקבל מקבל את החבילות ומאשר את הקבלה, הוא מקבל את החבילות רק לפי סדר שליחתן.

לאחר כל הגעת חבילה, הצד המקבל שולח ACK כי החבילה הגיעה, במקרה שהחבילה לא הגיעה לא נשלח ה-ACK . ברגע שהתקבל ACK על חבילה אצל השולח, השולח מזיז את החלון ביט אחד ימינה (לכיוון הביטים שטרם נשלחו) ושולח את החבילה הבאה, וכך מבצע כל פעם שמתקבל ACK על חבילה .

הדרך להתמודדות עם אובדן חבילות שלא הגיעו ליעדן בפרוטוקול זה היא בכך שקיים טיימר לשולח, השולח מחכה פרק זמן מסוים ואם בפרק הזמן שנקבע לא התקבל ACK , המאשר את הגעת החבילה לשולח-כך הוא יודע שהחבילה "נאבדה", ותתבצע שליחה מחודשת של כל החבילות אשר נמצאות בחלון.

כיצד המערכת מתגברת על בעיות Latency:

בעיות $latency$ הן בעיות השהייה ברשת, הכוונה להתייחסות בעיכוב בזמן שלוקח לקבל את החבילות ועד הזמן שמחזירים כי החבילות הגיעו.
נפרט על מספר סיבות אשר יכולות לגרום לבעיות מסוג זה:
1) מֶרְחָק- אחד הגורמים העיקריים להשהיית הרשת הוא המרחק, או כמה רחוק המכשיר שמגיש בקשות נמצא מהשרתים המגיבים לבקשות אלו.

2) בניית אתרים- הדרך שבה דפי אינטרנט נבנים משנה את זמן האחזור. דפי אינטרנט הנושאים תוכן כבד, תמונות גדולות או טוענים תוכן מכמה אתרי צד שלישי עשויים לפעול לאט יותר, מכיוון שדפדפנים צריכים להוריד קבצים גדולים יותר כדי להציג אותם.

3) בעיות של משתמש קצה-ייתכן שנראה כי בעיות רשת אחראיות להשהיה, אך לפעמים השהיית RTT היא תוצאה של כמות הזיכרון או מחזורי ה CPU-של המכשיר של משתמש הקצה נמוך כדי להגיב בפרק זמן סביר.

4)בעיות פיזיות-בהקשר פיזי, הסיבות השכיחות של השהייה ברשת הן הרכיבים שמעבירים נתונים מנקודה אחת לאחרת. כבלים פיזיים כגון נתבים, מתגים ונקודות גישה ל-WiFi. בנוסף, חביון יכול להיות מושפע ממכשירי רשת אחרים כמו מאזני עומסים של יישומים, התקני אבטחה, חומות אש ומערכות למניעת חדירות (IPS).

בשיטת המימוש שלנו $GO\ BACK\ N$ נשלחות חבילות בגודל חלון קבוע. הלקוח מקבל כל פעם חבילה אחר חבילה לפי הסדר. ועבור כל חבילה שנשלחה מחזיר ACK אל השולח(אצלנו זה השרת) כי החבילה הגיעה, השולח מזיז את החלון ושולח את ההודעה הבאה בתור.

לשולח יש $TIME$ OUT - שזה זמן קבוע עד שצריכה להתקבל תגובה על כך שהחבילה הגיעה אל הלקוח.

ברגע שACK לא מתקבל בזמן המוגדר ב $time\ OUT$ עקב בעיות השהייה ברשת, השולח שוב ישלח את כל החלון עם 7 החבילות אל הלקוח.

תיעוד באמצעות Wireshark

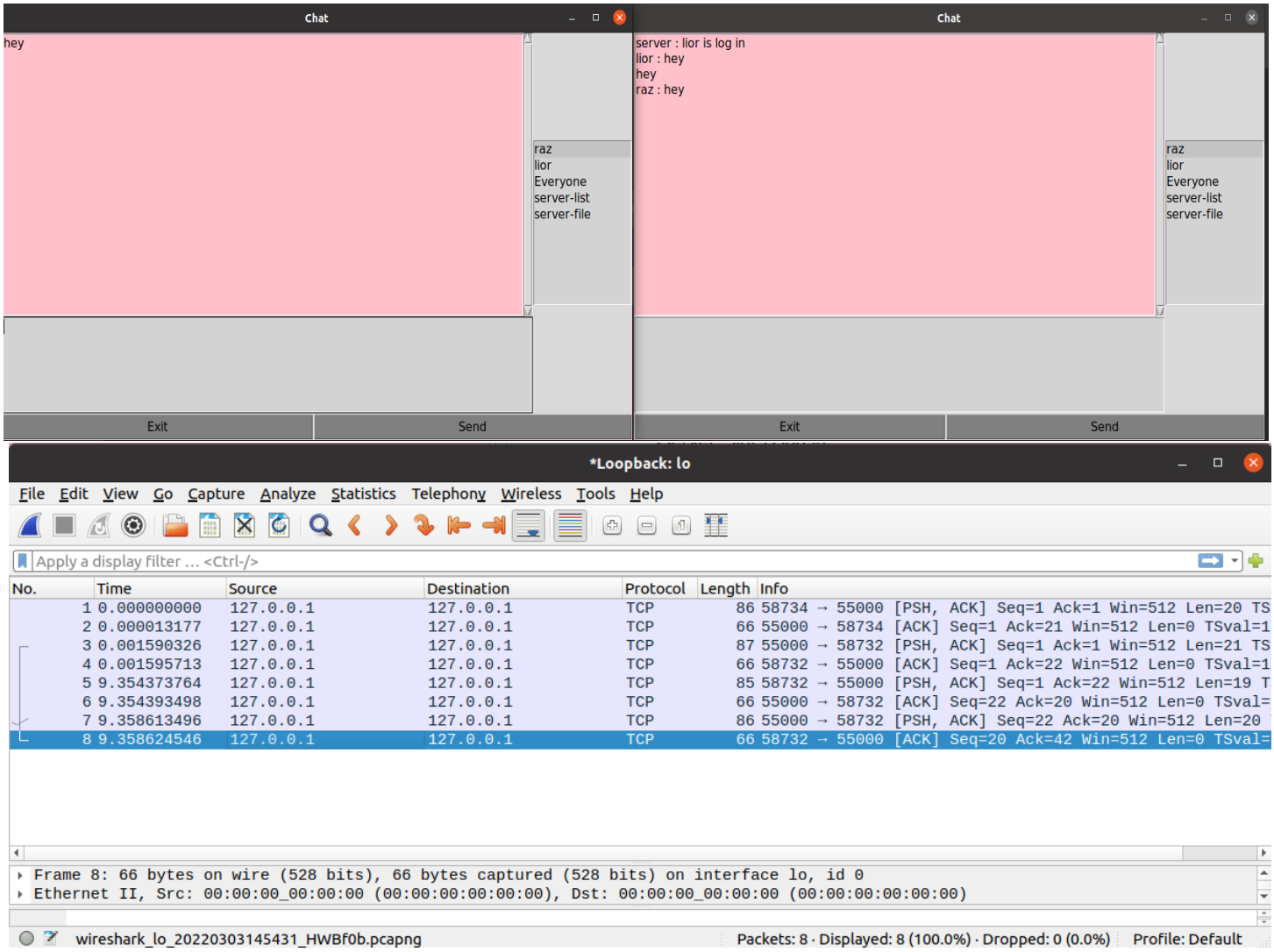
<u>כל התיעודים נמצאים תחת תיקייה בשם *wireshark* כאשר הניתוב יצוין בכל תיעוד שהבאנו בצילומי המסך הבאים:</u>

תקשורת בין שני לקוחות:

בחלק זה תיעדנו את החבילות העוברות ברשת כאשר משתמשים מדברים אחד עם השני, ניתן לראות שהפרוטוקול הוא פרוטוקול TCP כפי שהתבקשנו במטלה. כמו כן, ניתן לראות כי כתובת השליחה הינה 127.0.0.1 שהיא כתובת 'local host' שהיא הכתובת המקומית בכל מחשב.

מצורף למטלה :

קובץ ה*wireshark* בשם: *chat_users.pcapng*

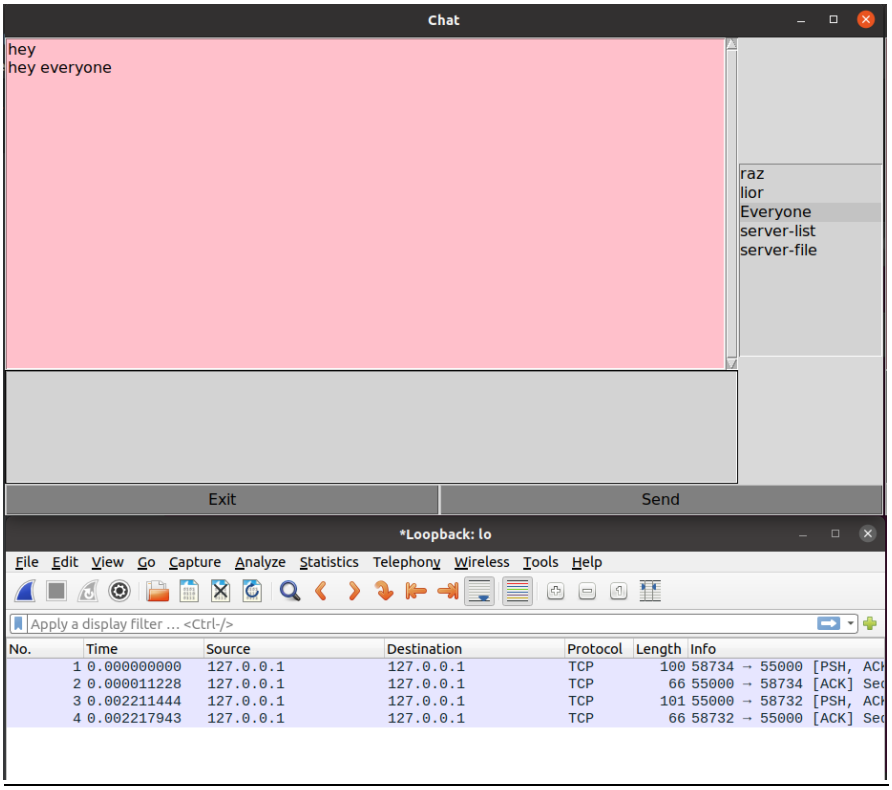


בס"ד

שליחת הודעה מלקוח אחד לכולם:

מצורף למטלה :

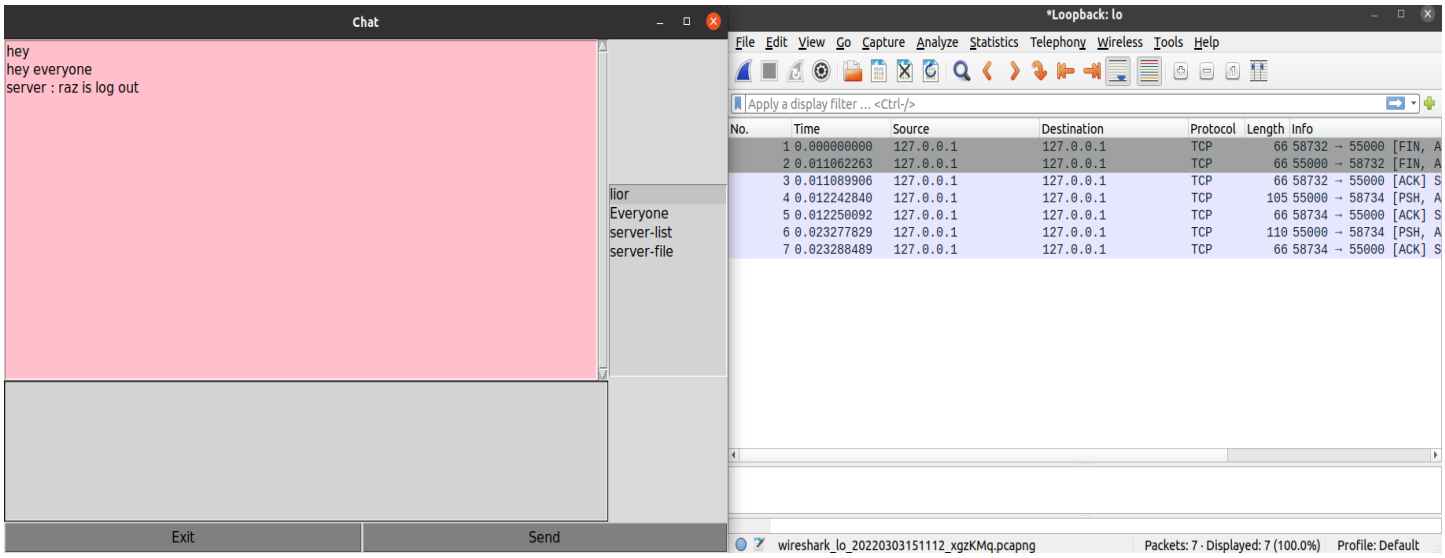
קובץ ה$wireshark$ בשם: $chat\_everyone.pcapng$



שליחת הודעת התנתקות של לקוח לכל הלקוחות:

מצורף למטלה :

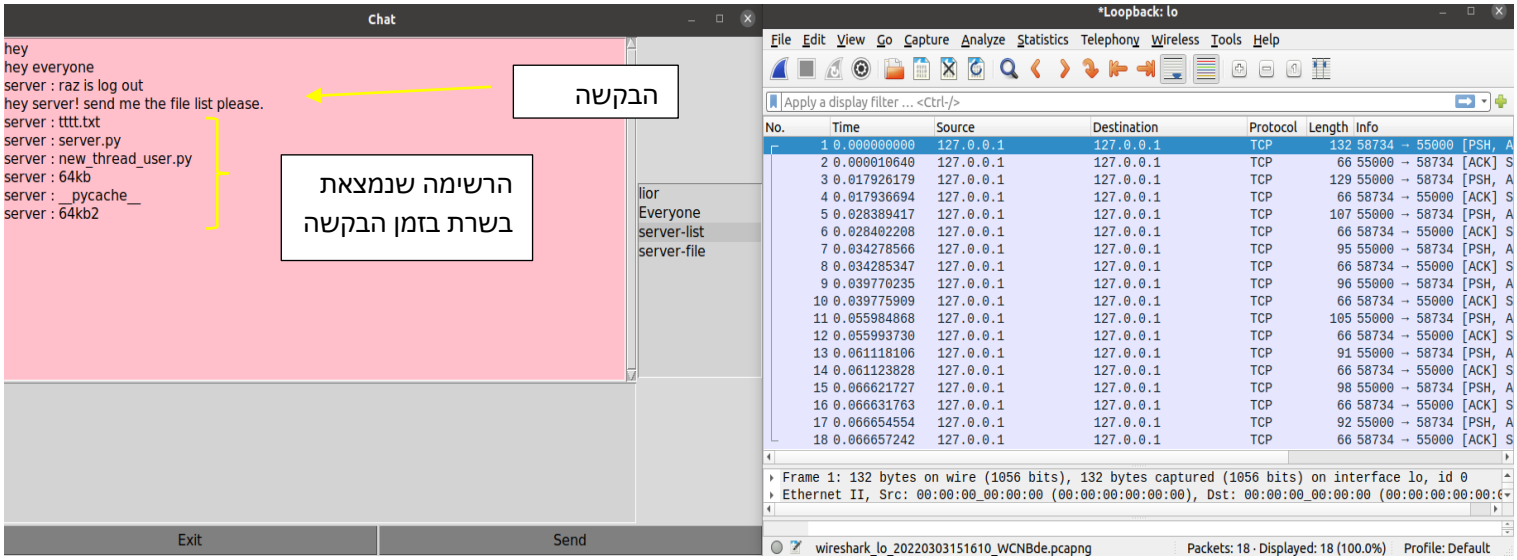
קובץ ה$wireshark$ בשם: $logout.pcapng$



בס"ד

תקשורת בין לקוח לשרת:

בחלק זה תיעדנו את החבילות העוברות ברשת כאשר לקוח יוזם בקשה להורדת קובץ מהשרת, ניתן לראות שהפרוטוקול הוא פרוטוקול TCP כפי שהתבקשנו במטלה. כמו כן, ניתן לראות כי כתובת השליחה הינה 127.0.0.1 שהיא כתובת 'local host' שהיא הכתובת המקומית בכל מחשב. נציין שוב כי ניתן לשלוח כל הודעה שנרצה למשתמש $server - list$ בכדי ליצור איתו קשר לקבל את רשימת הקבצים.

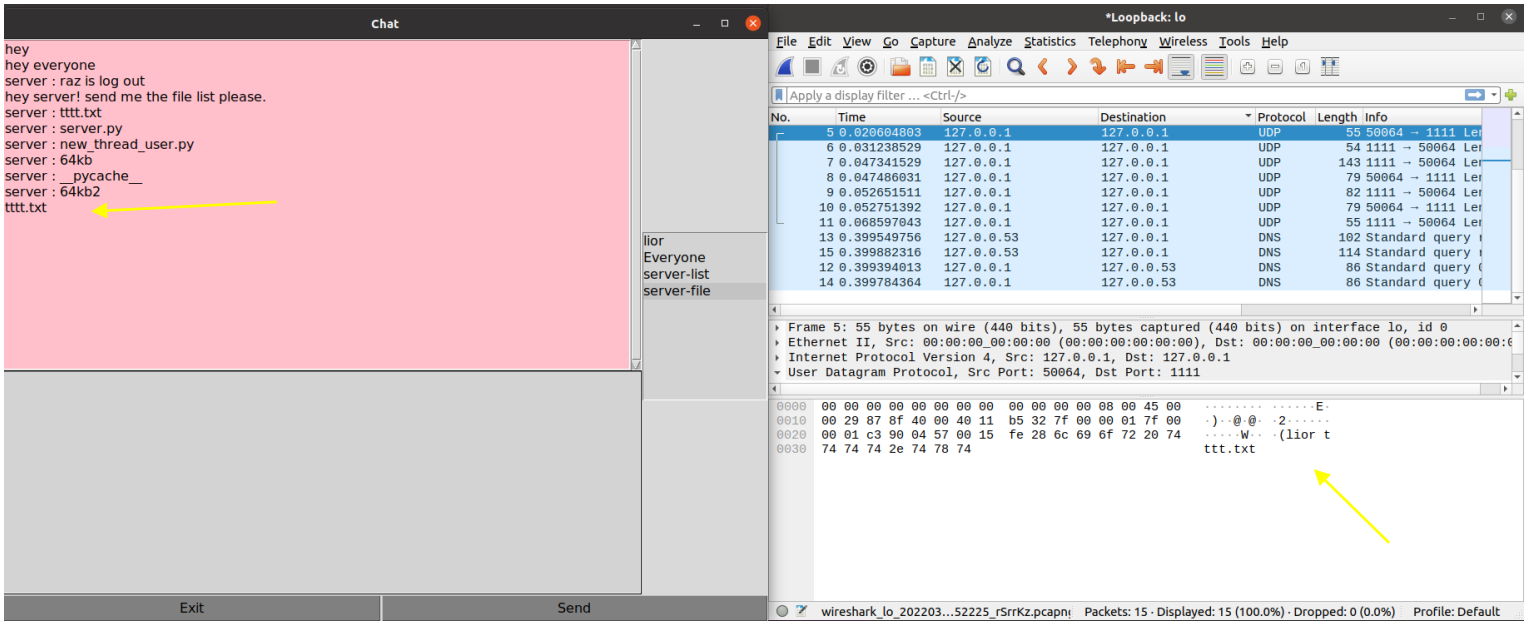


מצורף למטלה :

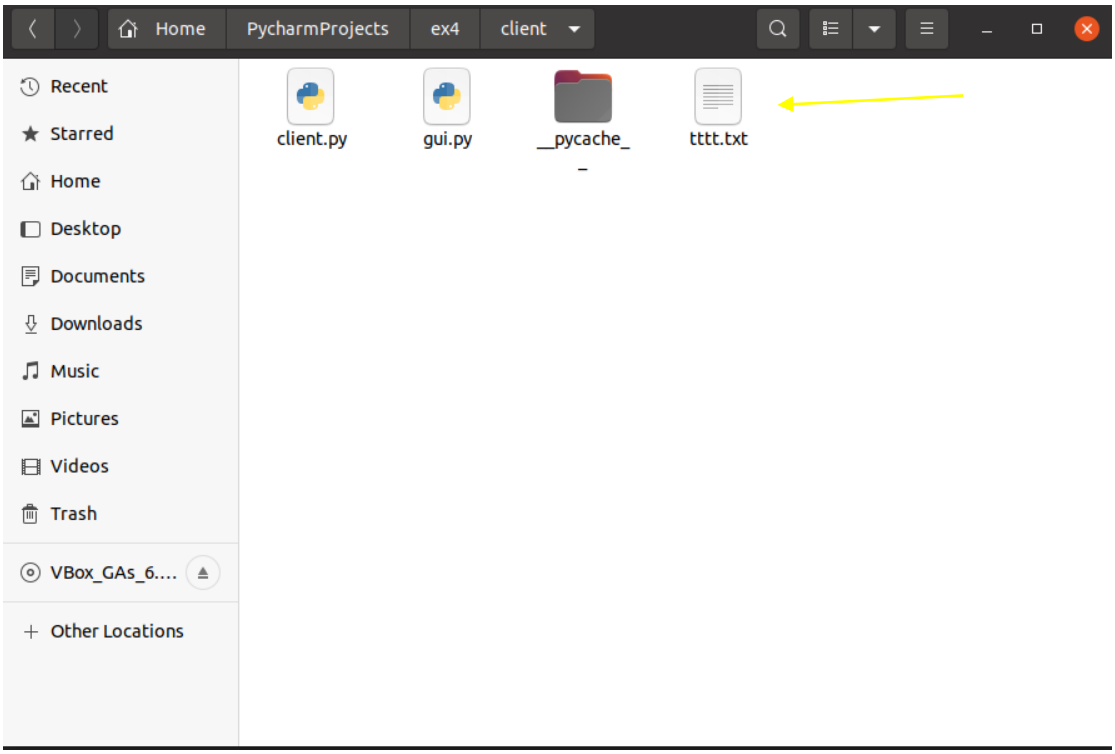
קובץ ה$wireshark$ בשם: $send\_list\_file.pcapng$

שליחת הקובץ מהשרת ללקוח:

ניתן לראות בצילומי המסך שהלקוח ביקש מהשרת את הקובץ tttt.txt , וקיבל אותו (ניתן לראות בצילומי המסך הבאים). כמו כן, ניתן לראות בתיעוד המובא כי פרוטוקול התקשורת של השליחה הינו פרוטוקול UDP כפי שהתבקשנו במטלה.



תמונה שהקובץ הועבר הקובץ הועבר בהצלחה:



מצורף למטלה :

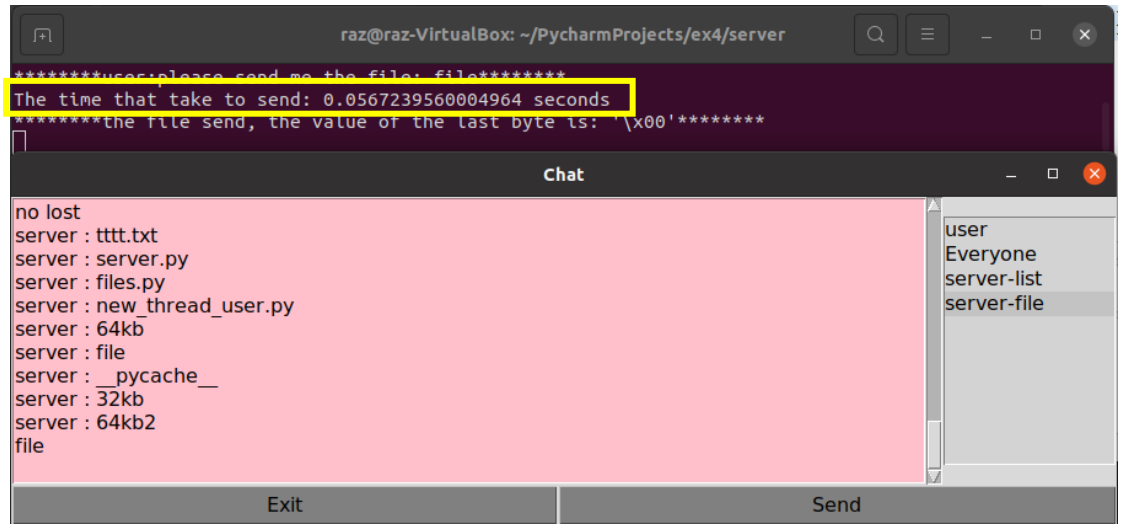
קובץ ה$wireshark$ בשם: $send\_file.pcapng$

יצירת איבוד חבילות

לפני יצירת איבוד חבילות: ניתן לראות שזמן השליחה הוא 0.056 שניות.

מצורף למטלה :

קובץ ה$wireshark$ בשם: $no\_lost.pcapng$



כעת, ניצור איבוד של פקטות, נציין את זמן השליחה שלקח בכל פעם ששלחנו את הקובץ file שהוא קובץ שיצרנו עבור המטלה ,קובץ זה נמצא בתיקייה אותה הגשנו עם הגשת המטלה.

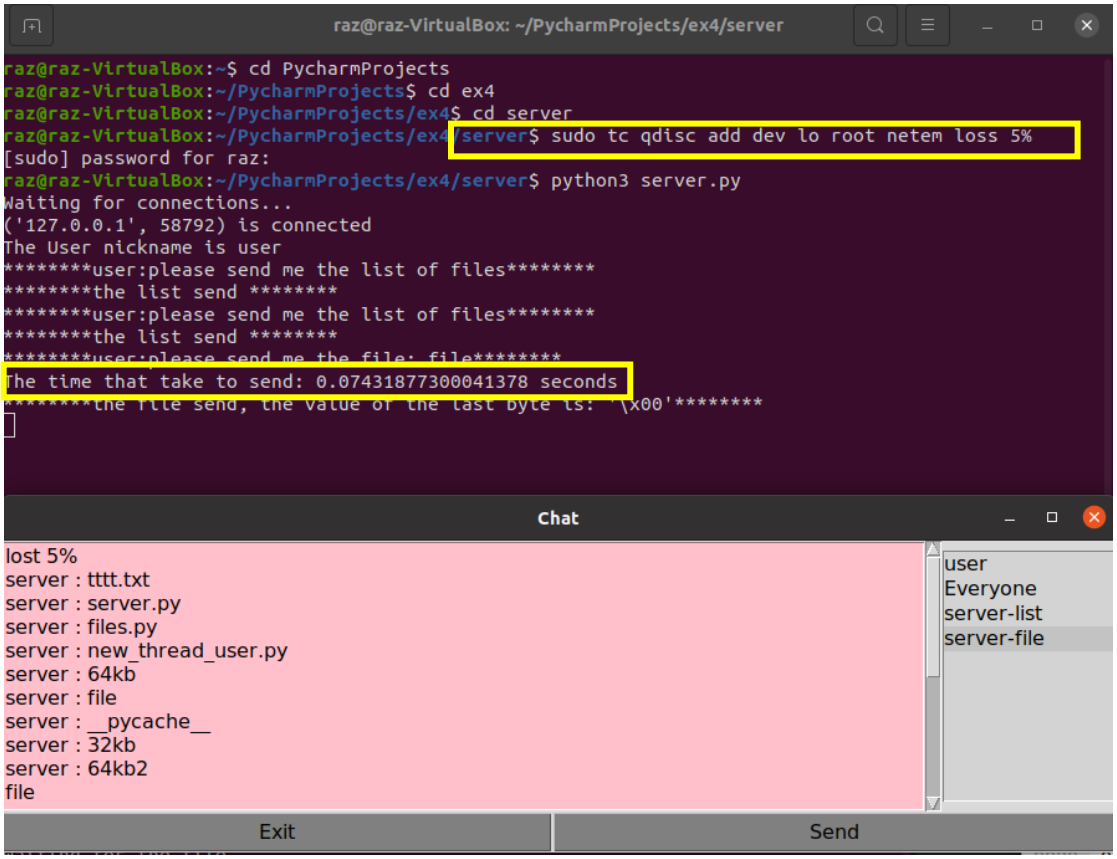
בס"ד

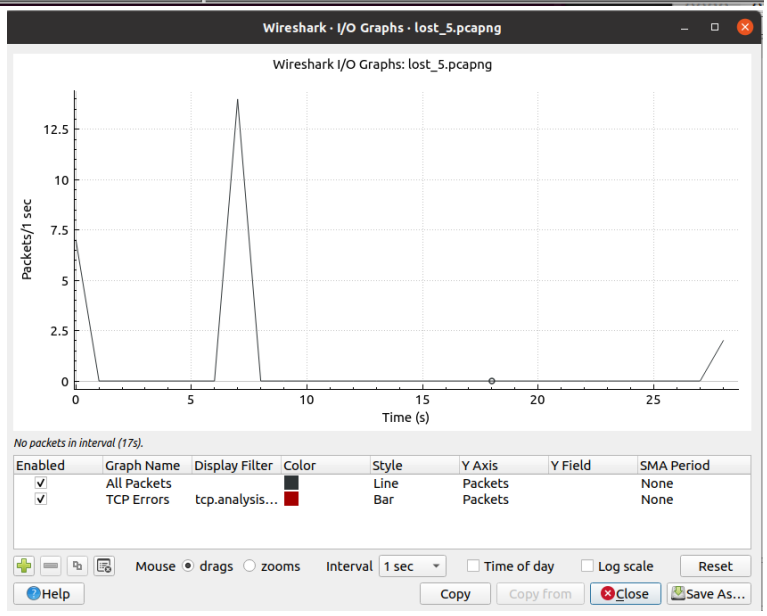
איבוד 5%:

בבדי ליצור איבוד של פקטות נצטרך להשתמש בפקודה הבאה :

sudo tc qdisc add dev lo root netem loss 5%

בה השתמשנו במטלה מספר 4 ליצור איבוד פקטות באופן רנדומלי, כמו כן ניתן לראות שבאשר המשתמש ביקש לשלוח את הקובץ הפעם, זמן השליחה שלו הוא 0.74 שניות בערך.





מצורף למטלה :

קובץ ה$wireshark$ בשם: $lost\_5.pcapng$

בס"ד

איבוד 10%:

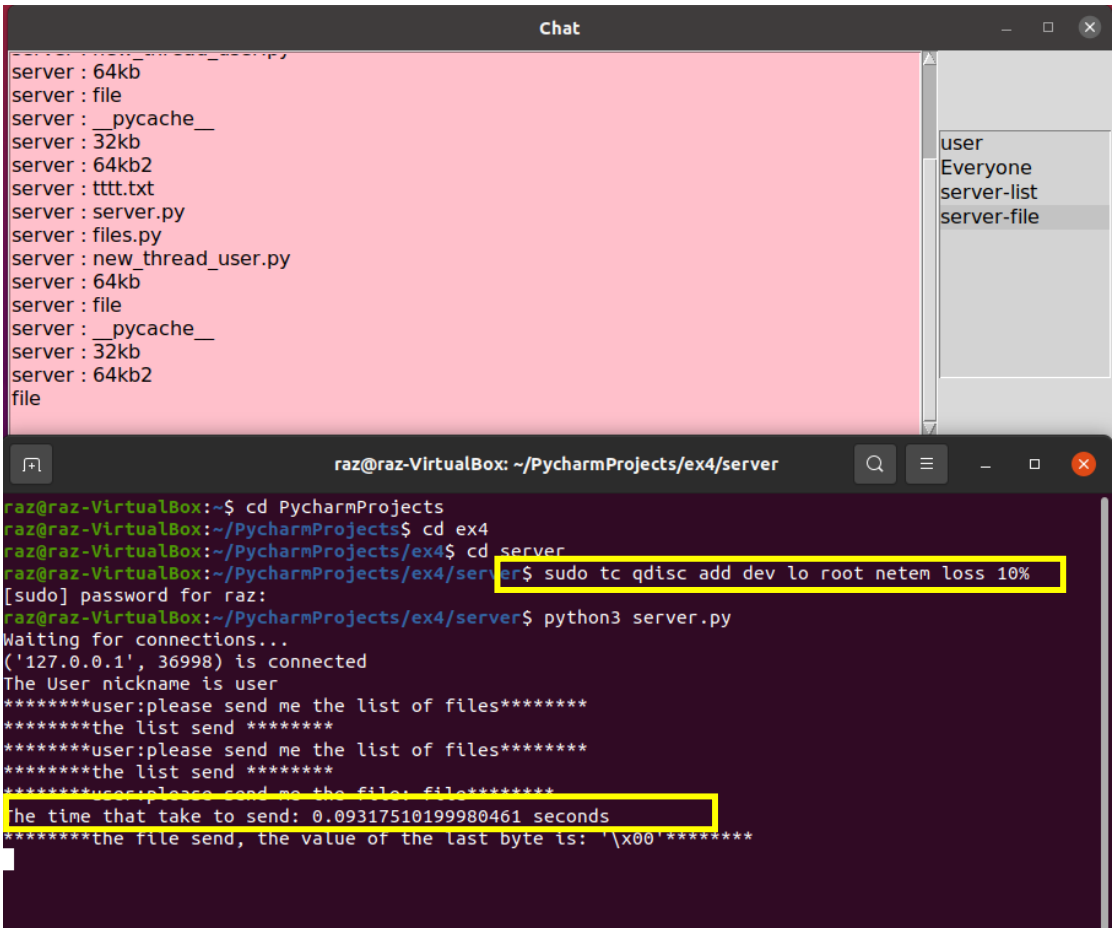
בכדי ליצור איבוד של פקטות נצטרך להשתמש בפקודה הבאה :

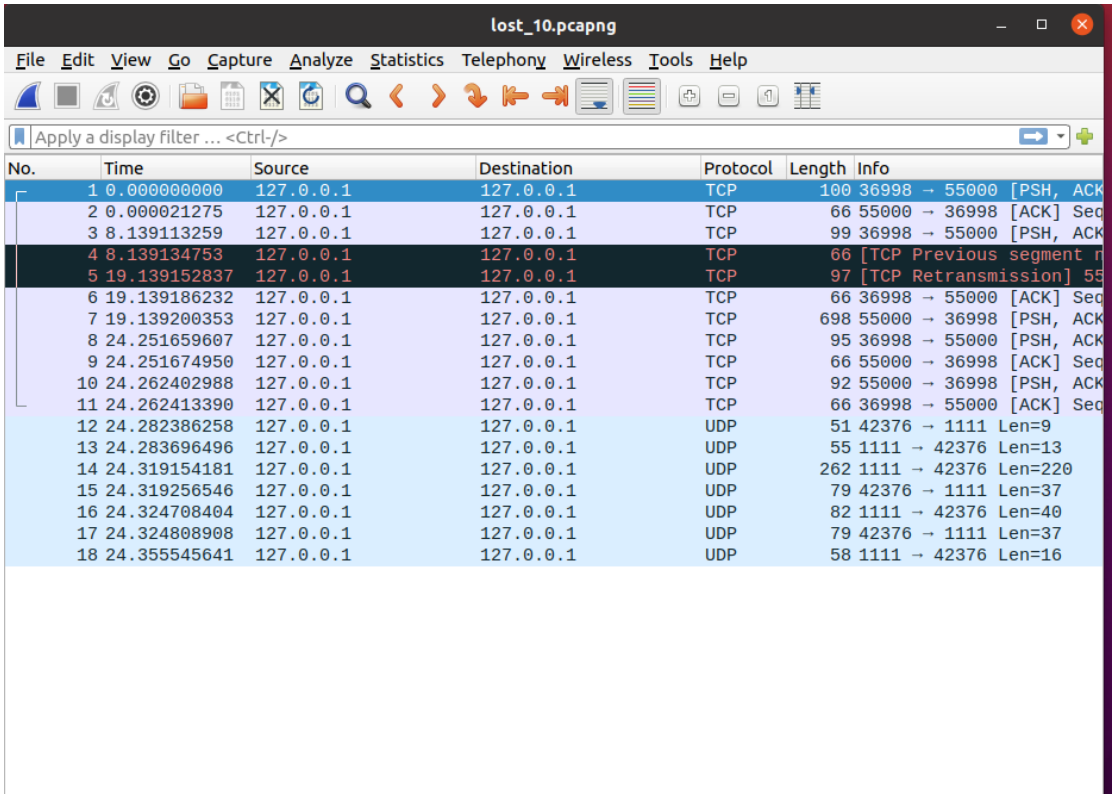
sudo tc qdisc add dev lo root netem loss 10%

ניתן לראות שכאשר המשתמש ביקש לשלוח את הקובץ הפעם, זמן השליחה שלו הוא 0.093 שניות בערך.

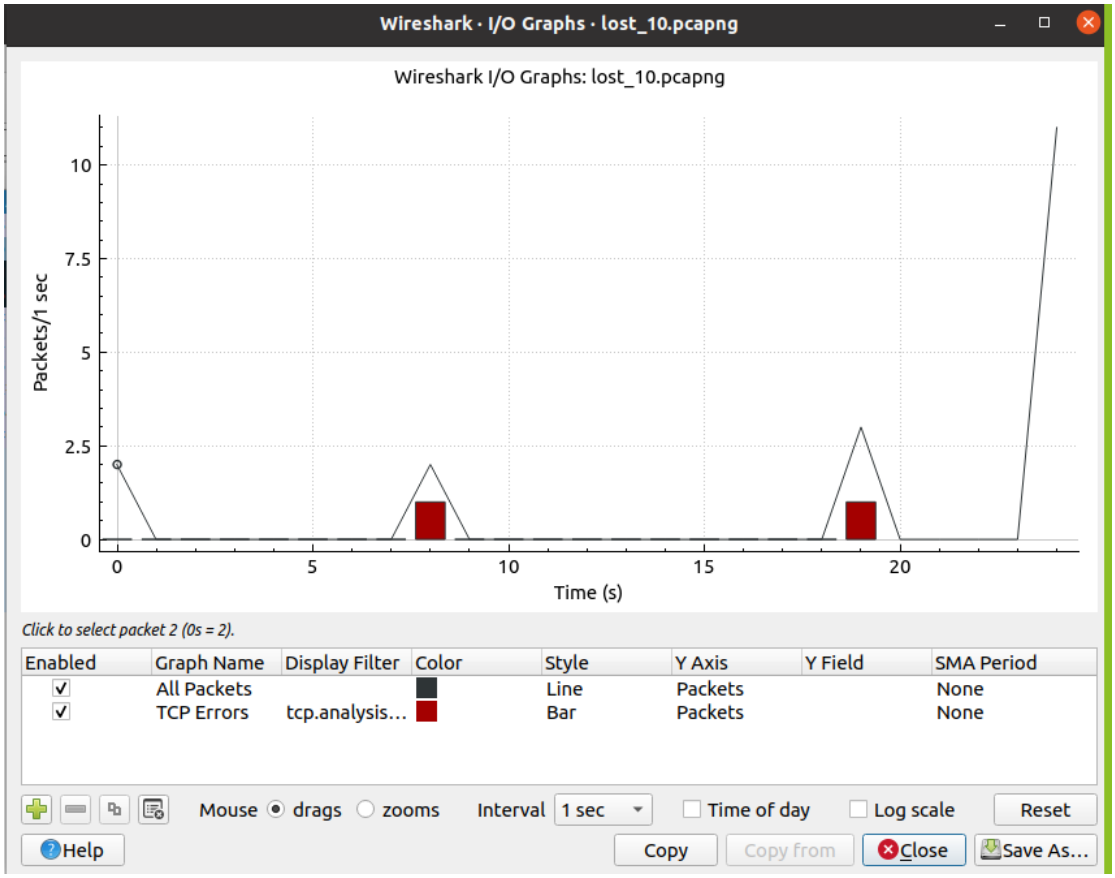
מצורף למטלה :

קובץ ה$wireshark$ בשם: $lost\_10.pcapng$





No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	127.0.0.1	127.0.0.1	TCP	100	36998 → 55000 [PSH, ACK
2	0.000021275	127.0.0.1	127.0.0.1	TCP	66	55000 → 36998 [ACK] Seq
3	8.139113259	127.0.0.1	127.0.0.1	TCP	99	36998 → 55000 [PSH, ACK
4	8.139134753	127.0.0.1	127.0.0.1	TCP	66	[TCP Previous segment n
5	19.139152837	127.0.0.1	127.0.0.1	TCP	97	[TCP Retransmission] 55
6	19.139186232	127.0.0.1	127.0.0.1	TCP	66	36998 → 55000 [ACK] Seq
7	19.139200353	127.0.0.1	127.0.0.1	TCP	698	55000 → 36998 [PSH, ACK
8	24.251659607	127.0.0.1	127.0.0.1	TCP	95	36998 → 55000 [PSH, ACK
9	24.251674950	127.0.0.1	127.0.0.1	TCP	66	55000 → 36998 [ACK] Seq
10	24.262402988	127.0.0.1	127.0.0.1	TCP	92	55000 → 36998 [PSH, ACK
11	24.262413390	127.0.0.1	127.0.0.1	TCP	66	36998 → 55000 [ACK] Seq
12	24.282386258	127.0.0.1	127.0.0.1	UDP	51	42376 → 1111 Len=9
13	24.283696496	127.0.0.1	127.0.0.1	UDP	55	1111 → 42376 Len=13
14	24.319154181	127.0.0.1	127.0.0.1	UDP	262	1111 → 42376 Len=220
15	24.319256546	127.0.0.1	127.0.0.1	UDP	79	42376 → 1111 Len=37
16	24.324708404	127.0.0.1	127.0.0.1	UDP	82	1111 → 42376 Len=40
17	24.324808908	127.0.0.1	127.0.0.1	UDP	79	42376 → 1111 Len=37
18	24.355545641	127.0.0.1	127.0.0.1	UDP	58	1111 → 42376 Len=16



יצירת השהיות

בכדי ליצור השהיות, יצרנו משתמש בשם latency שתפקידו להיות מונה ולספור כל 6 ack שנשלחים, ובפעם ה-6 ליצור השהייה של שליחת תגובת ack על קבלת החבילה , כ-0.02 שניות על ידי הוספת שינה של הזמן שציינו, ובכך יצרנו השהייה . בחרנו במספרים 0.02 ו-6 באופן רנדומלי, כמו כן יכולנו ליצור השהייה אחרי כל מספר אחר שנרצה.

להלן צילומי המסך של הוספת המשתנה ליצירת ההשהיה:

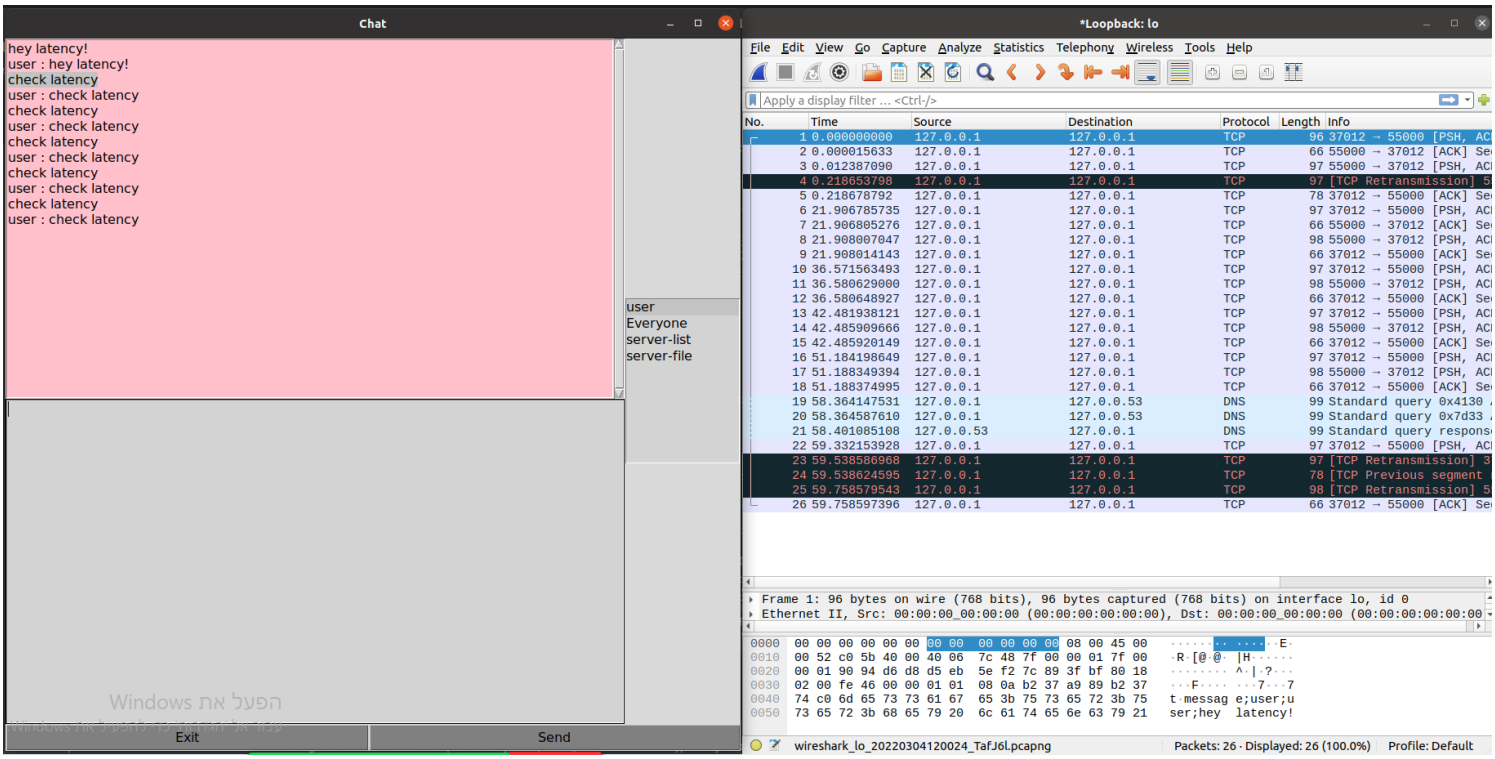
```
client.py
#create latncey
latency=0
```

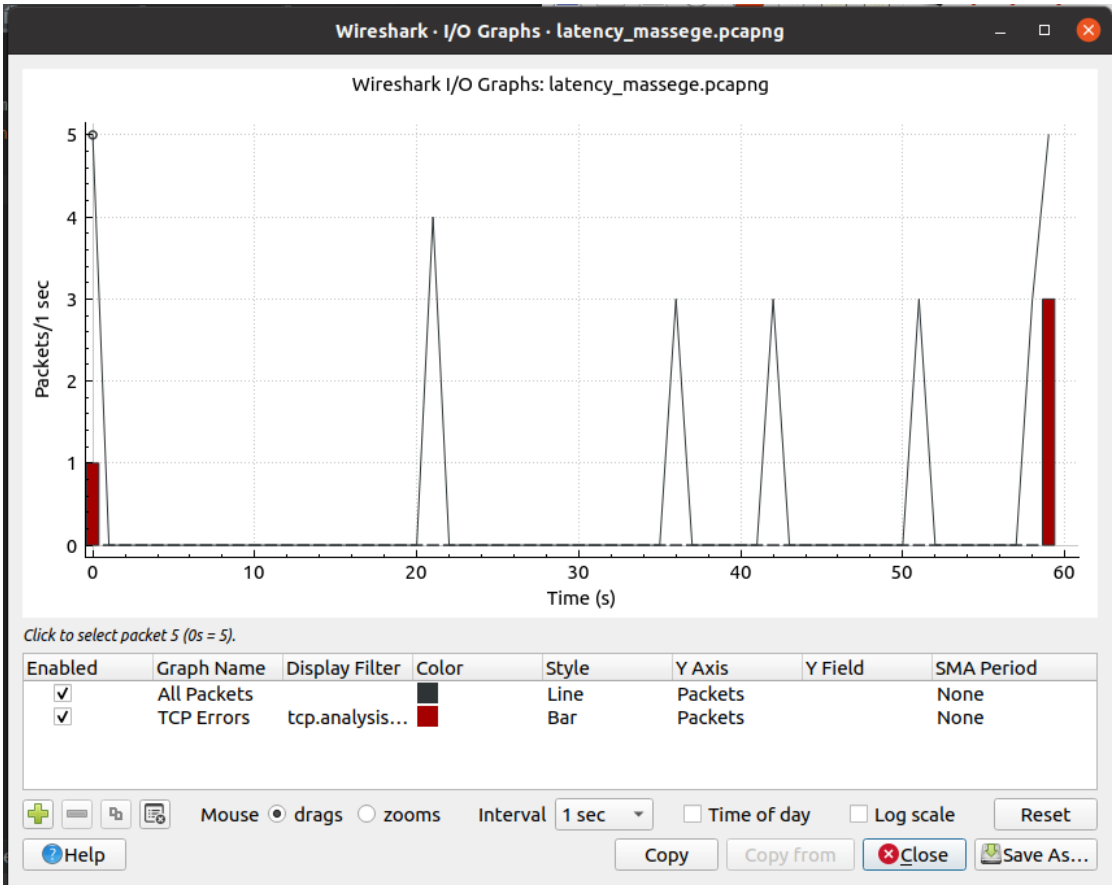
```
#create the latency
if latency==5:
    time.sleep(0.02)
    latency=0

# send to the server the ack
self.socket_file.sendto(pickle.dumps(send_packet), (client_address[0], client_address[1]))
latency += 1
```

יצירת השהייה ותיעוד בשליחת הודעה:

להלן צילומי מסך של ה*wireshark*: ניתן לראות שבשליחה של ההודעה מספר 6 אכן נוצרה השהייה בתיעוד.



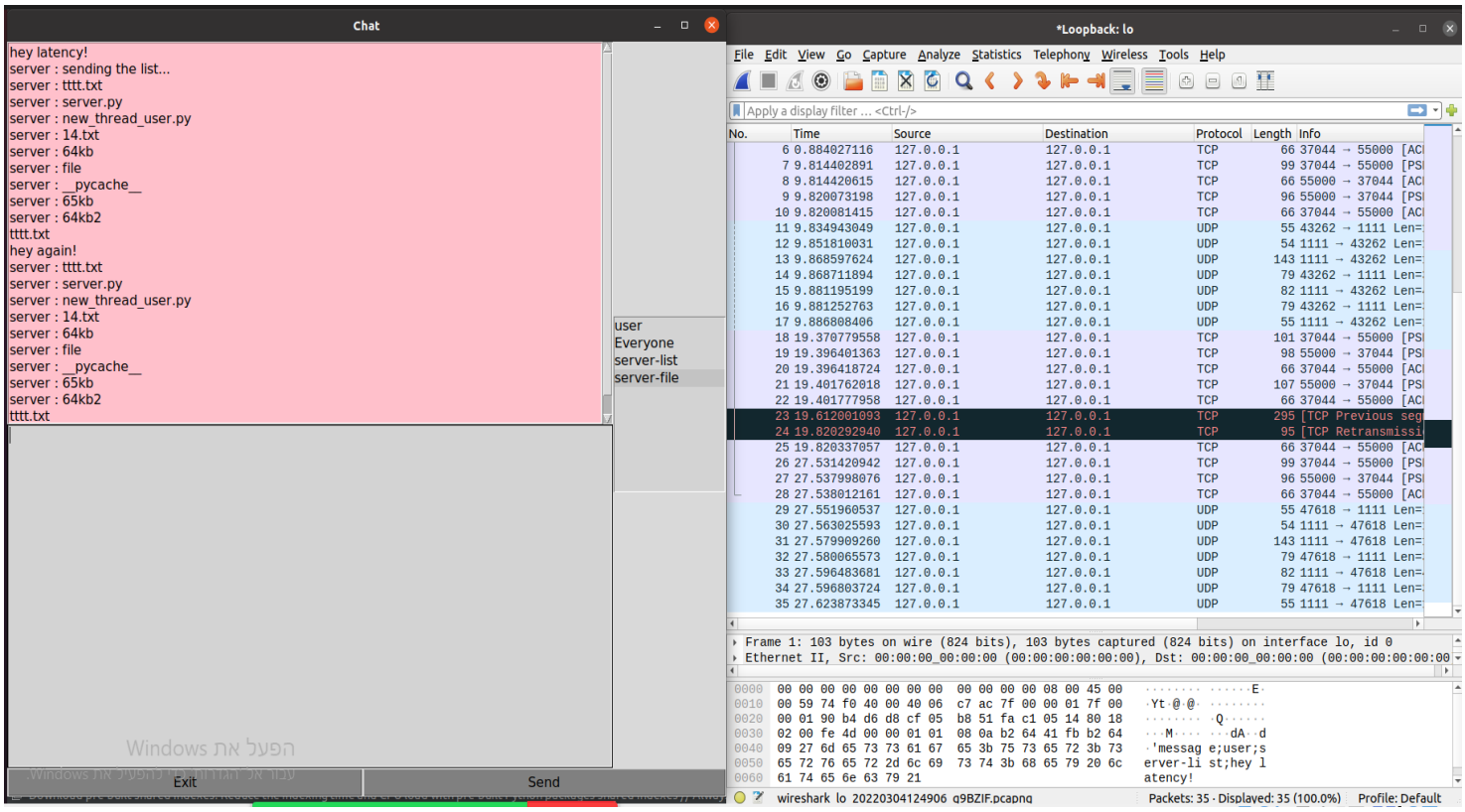


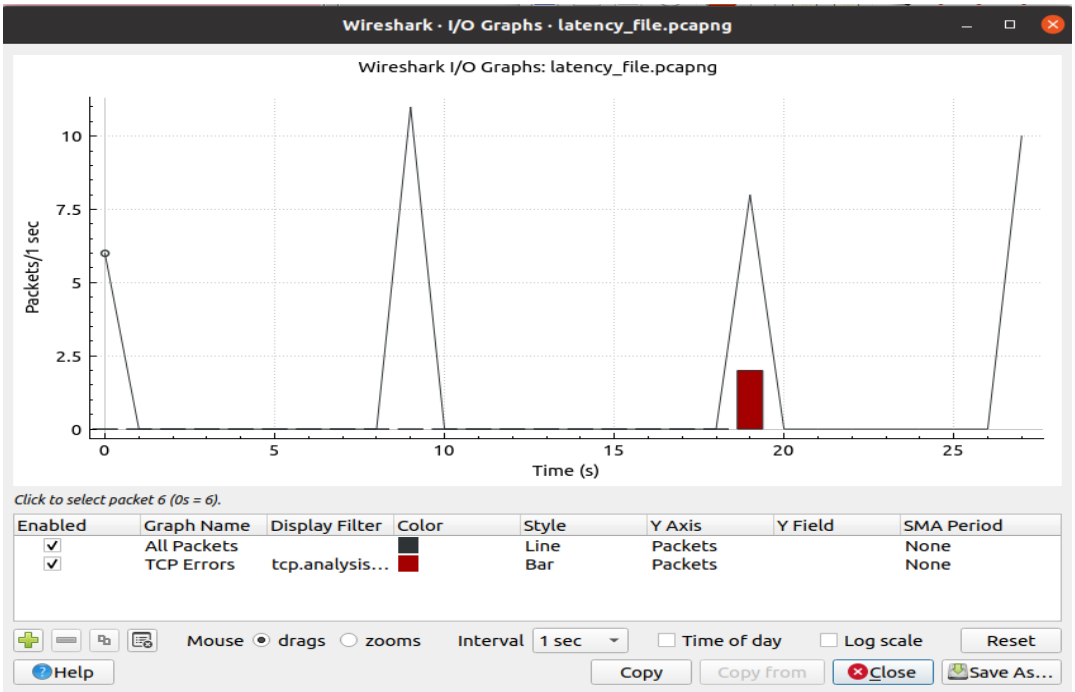
מצורף למטלה :

קובץ ה$wireshark$ בשם: $latency\_massege.pcapng$

בס"ד

יצירת השהייה ותיעוד בשליחת קובץ:





מצורף למטלה :

קובץ ה$wireshark$ בשם: $latency\_file.pcapng$

חלק ג'

1 .בהינתן מחשב חדש המתחבר לרשת אנא תארו את כל ההודעות שעוברות החל מהחיבור הראשוני ל switch ועד שההודעה מתקבלת בצד השני של הצאט. אנא פרטו לפי הפורמט הבא:

a .סוג הודעה, פירוט הודעה והשדות הבאים

i .כתובת IP מקור/יעד, כתובת פורט מקור/יעד, כתובת MAC מקור/יעד,

פרוטוקול שכבת התעבורה.

פתרון שאלה 1:

בהינתן מחשב חדש המתחבר לרשת יתבצע התהליך הבא:
ראשית, נצטרך לחבר את המחשב לרשת, בכדי לעשות זאת ישנן שתי דרכים אפשריות, הראשונה היא לחבר את המחשב באמצעות כבל רשת. השנייה היא חיבור באמצעות רשת אלחוטית. חיבור LAN קווי או אלחוטי מאפשר לגלוש באינטרנט וברשת הביתית של אותו מחשב.
היות והאינטרנט הוא רשת עולמית של מחשבים, לכל מחשב המתחבר לרשת בין אם הוא חדש או ישן חייבת להיות כתובת ייחודית הנקראת כתובת IP המחלוקת לארבעה חלקים X.X.X.X כאשר כל X הינו מספר בין 0 ל255.
כתובת ה-IP משמשת אותנו בתקשורת בין שני המחשבים השונים בכך שהיא מייחדת כל אחד מהם.

כל מחשב, מחובר לרכיב תקשורת הנקרא ראוטר (נתב) אשר נועד לקבוע את הנתיב, כשמו כן הוא, של חבילות נתונים ברשתות תקשורת נתונים. כאשר נתב מקבלת חבילת נתונים כלשהי, הוא בודק את כתובת היעד של החבילה ומנתב אותה ליעדה על סמך טבלת ניתוב המחוזקת במסד נתונים בנתב וכוללת את כל הניתובים האפשריים לשינוע חבילות ברשת, כמובן באופן יעיל ביותר על סמך שיטת מספר הצעדים הקטן ביותר.
נתב הוא switch של חבילות, ותפקידו העיקרי הוא ליצור גשר בין הרשת הביתית לרשת העולמית.
switch(מתג)הוא רכיב נוסף ברשת המחבר בין מספר צמתים שונים ברשת. למתג יש כניסות, והוא מכיר בכל כתובות ה-MAC של הרכיבים המחוברים לכל כניסה שלו על ידי כך שהוא מחזיק בטבלה שנקראת טבלת MAC. כאשר חבילה מגיע אל המתג, הוא מבצע בדיקה של כתובת ה-MAC אליה החבילה מיועדת-במידה ומכיר אותה הוא יעביר את החבילה אל הכניסה המתאימה, במידה ולא מכיר אותה המתג יעביר את החבילה לכל הכניסות מלבד הכניסה שממנה החבילה נכנסה.

כעת, לאחר שהסברנו איך חבילות עוברות ברשת כאשר מחשב חדש מתחבר נוכל להסביר באופן כללי איך מתבצע הליך שליחת הודעה ממחשב המקור אל מחשב היעד.
ההודעה אשר נכתבת בשכבת האפליקציה תתחיל לעבור בערמת הפרוטוקולים במחשב, כאשר ההודעה נשלחת, היא תפורק לחלקים קטנים יותר בכל שכבה-חלקים אלו מכונים מנות. לאחר שהחבילה עברה משכבת האפליקציה היא תגיע לשכבת ה-TCP, לכל מנה ישנו מספר פורט בכדי שהתוכניות ידעו היכן "להאזין" לחבילה ,כלומר, היכן לצפות לקבל אותה. השכבה הבאה שהחבילות יגיעו אליה היא שכבת ה-IP , בשכבה זו כל חבילה (מנה) מקבלת את כתובת ה-IP של היעד שלה. בשלב זה, כאשר לחבילות של ההודעה של המקור יש כתובת IP ופורט, הן הגיעו לשלב שבו ניתן לשלוח אותן דרך האינטרנט. השכבה הבא בה החבילות יעברו הינה שכבת החומרה שתפקידה לקחת את חבילות ההודעה ולהפוך את הטקסט בה לאותות אלקטרוניים. למחשב המקור יש ספק אינטרנט, הנתב של אותו ספק אינטרנט בוחן את כתובת ה-IP של היעד וקובע היכן לשלוח אותן. לאחר שנקבע לאן לשלוח את החבילות, החבילות עוברת לנתב אחר. כאשר הן מגיעות אליו, הן מתחילות לעלות במעלה השכבות בסדר הפוך לסדר שליחתן ממחשב המקור. הן יתקבלו בשכבת החומרה של מחשב היעד , יעלו לשכבת ה-IP , לאחר מכן לשכבת ה-TCP ואז החבילות יורכבו

בחזרה לכדי ההודעה אשר תוצג במחשב היעד, וזו בדיוק ההודעה שנשלחה ממחשב המקור. יש לציין, כפי שהסברנו ההודעה הנשלחת ממחשב המקור מפורקת לחבילות שכן אחת מהן מקבלת כתובת ופורט, כאשר החבילות עולות כלפי מעלה, כל נתוני הניתוב אשר נוספו על ידי מחסנית המחשב השולח נמחקים מהמנות.

2 .הסבירו מה זה CRC

פתרון שאלה 2: ראשי תיבות של Cyclic Redundancy Check זהו קוד לזיהוי שגיאות המשמש כדי לקבוע אם קיימות שגיאות בהעברת נתונים, לפני העברת המידע מחושב ה־CRC ומתווסף למידע המועבר.
לאחר העברת המידע, הצד המקבל מאשר באמצעות ה־CRC שהמידע הועבר ללא שינויים.

אופן הפעולה
CRC מבוסס על פולינומים, כך שמסתכלים על כל וקטור באורך n כפולינום שמקדמיו הם קואורדינטות הווקטור.
CR משתמש בפולינום המוגדר בפולינום יוצר מדרגה r, בהינתן פולינום כנ"ל ובהינתן הודעה שברצוננו לקדד, עלינו לבצע את הפעולות הבאות:
ראשית, נוסיף r אפסים מימין להודעה. שנית, נחלק בפולינום (תוך שימוש בחילוק של השדה מודולו 2)ונחסר את השארית תוך שימוש בשער לוגי xor.לבסוף, נצרף את התוצאה שקיבלנו מימין להודעה המקורית ונשלח.
כמו בכל קידוד Checksum, הצד המקבל יבצע את שלבים 1 ו-2 ויוודא ש-r הביטים האחרונים שנשלחו זהים לתוצאה שהתקבלה.

3.מה ההבדל בין http 1.0 ,http 1.1, http 2.0, QUIC?
פתרון שאלה 3:
HTTP/1.0 תוכנן להשתמש בחיבור TCP חדש עבור כל בקשה, לכן כל בקשה סבלה מהעלות של הקמת חיבור TCP חדש. בעוד ש-HTTP /1.1 מאפשר שימוש חוזר בחיבור.

HTTP/ 1.1 טוען משאבים בזה אחר זה, כך שאם לא ניתן לטעון משאב אחד, הוא חוסם את כל המשאבים האחרים, כלומר אפשר לשלוח בכל פעם בקשה בודדת של תקשורת TCP בקשה נוספת יכולה לקרות רק לאחר שהתקבלה תגובה לבקשה הנוכחית. לעומת זאת, HTTP/2 מסוגל להשתמש בחיבור TCP יחיד כדי לשלוח זרמים מרובים של נתונים בו-זמנית, כך שאף משאב לא חוסם משאב אחר. HTTP/2 עושה זאת על ידי פיצול נתונים להודעות קוד בינארי ומספור הודעות אלו כך שהלקוח יודע לאיזה זרם שייכת כל הודעה בינארית.

HTTP/2 מהיר ויעיל הרבה יותר מ-HTTP/1.1. אחת הדרכים שבהן HTTP/2 מהיר יותר היא האופן שבו הוא נותן עדיפות לתוכן במהלך תהליך הטעינה. ב-HTTP/2, למפתחים יש שליטה מעשית ומפורטת על תיעדוף של איזו הודעה תישלח קודם. זה מאפשר להם למקסם את מהירות טעינת הדפים הנתפסת והממשית במידה שלא הייתה אפשרית. ב-HTTP/2, כאשר לקוח מגיש בקשה לדף אינטרנט, השרת שולח מספר זרמים של נתונים ללקוח בו-זמנית, במקום לשלוח דבר אחד אחרי השני. שיטה זו של מסירת נתונים ידועה בשם ריבוי. מפתחים יכולים להקצות לכל אחד מזרמי הנתונים הללו ערך משוקלל שונה, והערך אומר ללקוח איזה זרם נתונים להציג ראשון.

4 .למה צריך מספרי port?

פתרון שאלה 4: מספרי port הם מספרים באורך עד 16 ביטים, מהם ניתן ליצור 2^{16} (65536) כתובות אפשריות, אשר נועדו לזהות כתובות/פרוטוקולים כלשהם והמספור השונה נועד לשמור על העקביות.
השימוש השכיח ביותר במספרים אלו הוא בשכבת התעבורה של תקשורת מחשבים בכך שהפורט מאפשר העברת מידע מתוכניות בין מחשבים, ברשת משותפת או נפרדת, ולשם כך המספור הכרחי.

5 .מה זה subnet ולמה צריך את זה?
פתרון שאלה 5:

subnet- רשת משנה היא חלוקה של רשת IP, כאשר רשת IP היא קבוצה של פרוטוקולי תקשורת המשמשים באינטרנט וברשתות דומות אחרות. רשת משנה הוא כינוי לפעולת החלוקה, ובכדי לעבור בין רשתות משנה נעשה שימוש במכשירים המשמשים לחילוף תעבורה בין רשתות משנה שונות הנקראים נתבים. כתובת IP מורכבת ממספר רשת ומזהה מארח, ובכדי לזהות את רשתות המשנה השונות לכל המארחים ברשת משנה יש את אותה קידומת רשת, ואילו למזהה המארח יש זיהוי מקומי ייחודי רק לו. היתרון שבה הוא שבאמצעותה תעבורת הרשת יכולה לעבור מרחק קצר יותר ליעד שלה מבלי לעבור דרך נתבים מיותרים.

6 .למה צריך כתובות mac למה לא מספיק לעבוד עם כתובות ip?
פתרון שאלה 6: בעת הייצור של כל רכיב תקשורת מוטבע מזהה ייחודי המורכב משדות של מספרים ואותיות המוטבע על כל רכיב תקשורת בעת הייצור הנקרא כתובת MAC: Media Access Control address .
בכדי לתקשר בין 2 מחשבים או יותר, יש צורך בכתובת ייחודית לכל מחשב ולכל התקן תקשורת.
הסיבה שלמחשבים יש גם כתובות MAC וגם כתובות IP היא שכתובות MAC מטפלות בחיבור הפיזי ממחשב למחשב בעוד שכתובות IP מטפלות בחיבור הלוגי הניתן לניתוב הן מהמחשב למחשב והן מהרשת לרשת.

7. מה ההבדל בין Router Switch Nat ?

פתרון שאלה 7: כאשר התקשורת נעשית בתוך הרשת הביתית אין צורך בשימוש בראוטר, נהיה זקוקים לראוטר רק שנרצה לחבר את הרשת הביתית לאינטרנט או לרשת אחרת חיצונית.
הסוויץ מעביר תקשורת בין רכיבי התקשורת באותה רשת , תקשורת זו מבוצעת ע"י כתובת MAC address .
לכל סוויץ יש טבלה שבה יש שיוך בין פורט פיזי לבין כתובת MAC address , הסוויץ לומד את כתובות ה MAC ושומר אותם בטבלה. לעומת זאת, כאשר הרשת הפנימית שלנו צריכה להתחבר לאינטרנט היא חייבת להשתמש ברואטר.
פנייה של מחשב מתוך הרשת הפנימית שלנו למחשב אחר שמחובר לרשת האינטרנט נעשית בדרך הבאה : קודם, מידע מכתובת פנימית אשר מגיעה לרואטר נשלחת עם כתובת IP רגילה. לאחר מכן, הראוטר צריך לנתב את המידע למחשב או להתקן הפנימי, לשם כך משתמש בטבלה שנקראת NAT. טבלת ה-NAT ממירה את הכתובת הפנימית ל IP חיצוני , כתובת היעד ופורט היעד תמיד נשמרים גם שהמידע יעבור דרך ראוטרים אחרים הנמצאים ברשת האינטרנט.

8 .שיטות להתגבר על המחסור בIPv4 ולפרט?

פתרון 8 שאלה : מגבלת כמות הכתובות שהיו קיימות על-פי גרסה 4 של פרוטוקול ה-IP, מובילה למעבר אל גרסה 6 שלו, המאפשר לחלק יותר כתובות IP.

IP גרסה 4 היא כתובת IP היא בת 32 סיביות. תאורטית, שיטה זו מאפשרת עד 2 בחזקת 32 (מעל 4 מיליארד) כתובות שונות, אולם, מכיוון שטווחים גדולים של כתובות שמורים למטרות מיוחדות, מספר הכתובות השמישות קטן יותר.
בתחילה, כשהומצא ה-IPv4, ממציאיו לא העלו בדעתם שיהיו כל כך הרבה מכשירים שיצטרכו כתובות, כיוון שהרשתות אז לא היו נפוצות. ככל שהאינטרנט נהיה נפוץ הבעיה התחילה לצוף ולכן הוציאו את גרסת IPv6.

על מנת להתמודד עם מצוקת כתובות ה-IP שנוצרה הוגדר תקן חדש, IPV6 שבה כל כתובת IP מורכבת מ-8 קבוצות של 16 סיביות, כלומר 128 סיביות. תקן זה מאפשר מרחב עצום של 2 בחזקת 128 (מספר בעל 38 ספרות) כתובות שונות, ופותר את המחסור בכתובות.

9 .נתונה הרשת הבאה.

OSPF מריצים a. AS2, AS3

RIP מריצים b. AS1, AS4

c .בין ה-Ass רץ BGP

d .אין חיבור פיזי בין AS4, AS2

e .בעזרת איזה פרוטוקול לומד הנתב c3 על תת רשת x

f .בעזרת איזה פרוטוקול לומד הנתב a3 על תת רשת x

g .בעזרת איזה פרוטוקול לומד הנתב c1 על תת רשת x

h .בעזרת איזה פרוטוקול לומד הנתב c2 על תת רשת x

פתרון שאלה 9:

ראשית נסביר את התהליכים המופעלים:

OSPF- Open Shortest Path First הוא פרוטוקול ניתוב ברשתות שמתבסס על כתובת היררכית, תלוי מצב, להעברת נתונים בין מספר ראוטרים באותה מערכת. פרוטוקול זה משתמש באלגוריתם $Dijkstra$ (דייקסטרה), אשר פותר את בעיית מציאת המסלול הקל ביותר מנקודה בגרף ליעד בגרף מסוג עץ ממושקל. באמצעות אלגוריתם זה, הפרוטוקול מבצע חישוב של העברת הנתונים ובחירת הנתיב הזול ביותר בחבילה מהמקור ליעד.

RIP -Routing Information Protocol פרוטוקול קודם ל-OSPF הוא פרוטוקול ניתוב המתבסס על הכרת ורישום הנתבים והרשתות המחוברות אליהם, וכמות הצעדים בכל נתיב לכל יעד. כאשר חבילה מגיעה אל נתב כלשהו, באמצעות ספירת צעדים הוא מחפש את כמות הצעדים המינימלית אל היעד, בהתאם לרישומים שלו, ובכך ספירת צעדים תועיל לו בכדי לבצע החלטות ניתוב טובות יותר. בנוסף, הנתב מבקש לקבל עדכונים לגבי שינויים בנתיבים המובילים אל היעד בפרק זמן של 30 שניות, בכך הוא מקבל מידע על נתבים חדשים שחוברו לרשת ונשאר מעודכן בנוגע למתרחש בסידור הפיזי או הלוגי של הרכיבים השונים ברשת מחשבים.

BGP- Border Gateway Protocol הוא פרוטוקול ניתוב המהווה את ליבת מערכת הניתוב של רשת האינטרנט. פרוטוקול ניתוב זה מבוסס קשר המתפקד בשכבת היישום של מודל ה-TCP/IP וכן בשכבת היישום של מודל
ה־OSI -מודל המציג את הפעולות השונות הנדרשות על-מנת להעביר נתונים ברשת תקשורת, ואת הסדר בין הפעולות השונות. נתב אשר פועל באמצעות פרוטוקול זה מבצע רישום של הרשתות המחוברות, והקשרים בינו לבין רשתות אחרות, ועל סמך הקשרים הללו הוא מבצע החלטות ניתוב המוכתבות בצורה ידנית על ידי מנהל הרשת שתפקידו לנהל את האמצעים המרכיבים את רשת המחשבים של הארגון.

עמיתים של BGP מחליפים מידע על ניתוב על סמך חיבור TCP קבוע למחצה: שיחת BGP. מכיוון שBGP מספק לכל מערכת אוטונומית (AS) אמצעי לקבל תת רשת נגישה ומידע משכנים בעלי מערכת אוטונומית (AS), להפיץ מידע על נגישות לכל הנתבים הפנימים בכל מערכת אוטונומית (AS) ,ולקבוע נתיב טוב לתת רשת על בסיס מידע על נגישות ומדיניות.

על פי האמור לעיל, התשובה לשאלה זו היא:
כל הנתבים לומדים על תת הרשת x בכך שהם מייצר רשומה מתאימה ומכניסים אותה לטבלת הניתוב שלהם, באמצעות הרשומה הם יגיעו בדרך הקצרה ביותר x, כל אחד באמצעות השימושים בפרוטוקולים הבאים:

e.נתב c3 לומד על תת רשת x כך:
ראשית הוא צריך להשתמש ב פרוטוקול BGP בכדי להגיע למערכת האוטונומית AS4 , הנתב הראשון שיגיע אליו הוא 4c . שנית, על ידי פרוטוקול RIP ,שעל פי הנתונים זה הפרוטוקול שעובד בAS4 , הוא יוסיף לרשומה את הניתוב בדרך המהירה ביותר להגיע לתת הרשת x .

f.נתב a3 לומד על תת רשת x כך:
ראשית הוא צריך להגיע בדרך הקצרה ביותר לc3 על ידי OSPF שעל פי הנתונים זה הפרוטוקול שעובד ב AS3. לאחר מכן, באמצעות פרוטוקול BGP הוא יגיע למערכת האוטונומית AS4 , הנתב הראשון שיגיע אליו הוא 4c . לבסוף, על ידי פרוטוקול RIP שעל פי הנתונים זה הפרוטוקול שעובד בAS4 הוא יוסיף לרשומה את הניתוב בדרך המהירה ביותר להגיע לתת הרשת x .

g.נתב c1 לומד על תת רשת x כך:
ראשית הוא צריך להשתמש ב פרוטוקול BGP בכדי להגיע למערכת האוטונומית AS3, הנתב הראשון שיגיע אליו הוא a3 , הוא יגיע בדרך הקצרה ביותר לc3 על ידי שימוש בפרוטוקול OSPF שעל פי הנתונים זה הפרוטוקול שעובד בAS3 . לאחר מכן, באמצעות פרוטוקול BGP הוא יגיע למערכת האוטונומית AS4 , הנתב הראשון שיגיע אליו הוא 4c . לבסוף, על ידי פרוטוקול RIP שעל פי הנתונים זה הפרוטוקול שעובד בAS4 הוא יוסיף לרשומה את הניתוב בדרך המהירה ביותר להגיע לתת הרשת x .

h.נתב 2c לומד על תת רשת x כך:
היות ואין חיבור פיזי בין AS4,AS2 הוא צריך להגיע בדרך הקצרה ביותר לa2 על ידי OSPF שעל פי הנתונים זה הפרוטוקול שעובד בAS2 , ואז הוא צריך להשתמש בפרוטוקול BGP בכדי להגיע למערכת האוטונומית AS1, אליה הוא מחובר פיזית, הנתב הראשון שיגיע אליו הוא 1b. משם הוא יצטרך להוסיף לרשימה את הדרך להגיע לc1 שתושג על ידי פרוטוקול RIP שעל פי הנתונים זה הפרוטוקול שעובד בAS1 . לאחר מכן, הוא צריך שוב להשתמש ב פרוטוקול BGP בכדי להגיע למערכת האוטונומית AS3 הנתב הראשון שיגיע אליו הוא a3 ואז ימשיך לחפש את הדרך הקצרה ביותר לc3 על ידי פרוטוקול OSPF שעל פי הנתונים זה הפרוטוקול שעובד
בAS3 . ואז, בפעם השלישית ישתמש בפרוטוקול BGP בכדי להגיע למערכת האוטונומית AS4 , הנתב הראשון שיגיע אליו הוא 4c . לבסוף, על ידי פרוטוקול RIP שעל פי הנתונים זה הפרוטוקול שעובד בAS4 הוא יוסיף לרשומה את הניתוב בדרך המהירה ביותר להגיע לתת הרשת x .

לסיכום, ארבעת הנתבים 1c, 3a,3c ,2c לומדים על המערכות האוטונומיות אליהן הם מחוברים פיזית באמצעות פרוטוקול BGP, כאשר הניתוב בכדי להגיע לנתבים המקשרים בין על מערכת אוטונומית נעשה על ידי הפרוטוקולים בכל מערכת שלעצמה , ולבסוף יוצרים את המסלול היעיל ביותר להגעה לתת הרשת x, בכדי ללמוד עליה.

קישור לקובץ ליוטיוב

https://youtu.be/_eoWWujtyOU

תיעוד של השרת שהקבצים הועברו בהצלחה למשתמשים שביקשו אותם:

```
Waiting for connections...
('127.0.0.1', 56929) is connected
The User nickname is liam
('127.0.0.1', 56930) is connected
The User nickname is ori
********ori:please send me the list of files********
********the list send ********
********liam:please send me the list of files********
********the list send ********
********ori:please send me the file: 64kb********
********liam:please send me the file: 14.txt********
********the file send, the value of the last byte is: ' '********
The time that take to send: 1.1344180000014603 seconds
********the file send, the value of the last byte is: '\x00'********
The time that take to send: 4.061065499990946 seconds
The User ori has logged out.
The User liam has logged out.
```