מטלה 4 ברשתות תקשורת-

מגישים: יסמין כהן- 212733836
שלומי זכריה- 315141242

הקדמה-
במטלה הזאת למדנו איך להשתמש בsocket Raw וגם בפרוטוקול ICMP.
בהתחלה בנינו תוכנית ping.c בעזרת הקוד שהביאו לנו, מטרת ping.c היא לתקשר עם מכשיר מסוים שנתון לנו רק ה IP שלו , והתקשורת איתו נעשית בשליחת וקבלת ping שזה בעצם חבילות ICMP, ולכן בחיבור נשתמש בsocket raw.
בנינו בנוסף תוכנית better_ping.c שיש בה תוספת ל ping.c , התוספת היא watchdog שתפקידו להקציב זמן לקבלת הping, ולכן אם נכשל החיבור מאיזה שהיא סיבה או שלקח יותר מהזמן המותר לקבלת הping החיבור נסגר בצורה יפה. והתוכנית לא מחכה לנצח עד שיתקבל החבילה.
גם בתוכנית הזאת בכדי להתחבר לIP יעד (היעד שנרצה לקבל ממנו את הping) נשתמש בRaw Socket וכדי להתחבר לwatchdog (שמקציב לנו את הזמן) נשתמש בTCP socket.

:ping.c

דבר ראשון אנחנו פותחים Raw Socket מכוון שאנחנו עובדים עם פרוטוקול ICMP.
עבור כל פעולה, אם אירעה שגיאה, התוכנית תודיע מיד למשתמש ותיסגר.
בנינו חבילת ICMP:
חבילת ECHO REQUEST עם כל הפרמטרים והמידע שצריך, לדוגמא המספר הסידורי, ההודעה שנרצה לשלוח, והגודל של החבילה.
נשלח את החבילה בעזרת פונקציית sendto() אל כתובת IP של יעד.
אם מתקבלת תגובה, אז התוכנית תדפיס את חבילת ה-ICMP ECHO REPLAY שהתקבלה, היא תדפיס את: גודל החבילה, מספר סידורי שלה, ואת הזמן שלקח לה להתקבל.
אם לא התקבלה תגובה היא תחכה עד שיגיע, ואם לא יגיע היא תחכה לנצח.
אם התקבלה תגובה, התוכנית "תישן" שנייה אחת ואז שוב תחזור על אותה פעולה ותשלח עוד חבילה, עד שהמשתמש יחליט להפסיק (בעזרת control+c)
חשוב להבין שאם התוכנית לא מקבלת ping בחזרה היא תחכה לנצח.

Better Ping.c:

תוכנית Better Ping דומה לc.ping רק שיש בה הוספת שדרוג- השדרוג הוא watchdog שמודד את הזמן שלוקח לתגובה להגיע ומחליט מתי לסגור את התוכנית לפי זמן מוקצב מסוים, אצלנו זה 10 שניות, כדי שלא נחכה לנצח.
זה קורה באמצעות התוכנית "watchdog".
בbetter_ping דבר ראשון פותחים RAW socket שאנחנו רוצים שהוא יהיה במצב non-blocking, שזה אומר שבפונקציות שבדרך כלל מחכות זמן בלתי מוגבל עד שתהיה תגובה, לדוג' recv, הם כבר יצטרכו לחכות את כל הזמן , כך שפשוט אם לא התקבל התגובה אליה ציפו, הם יעברו לפקודה הבאה בקוד.
דבר שני נפתח גם TCP socket , כדי לתקשר עם הwatchdog.
בכדי ליצור ריצה בו זמנית של שתי התוכניות הbetter_ping והwatchdog נרצה להשתמש בthread, ולכן נשתמש בפונקציה $fork$ כדי ליצור תהליך נוסף שהוא watchdog.
ולכן כאשר אנחנו מריצים את better_ping זה מפעיל את תוכנית ה Watchdog בזמן ריצה.

כאשר נפתח תוכנית הwatchdog ,
דבר ראשון נפתח אצלו חיבור TCP socket והוא מאזין מעל פורט 3000 .
(אנחנו כאילו מתייחסים בתוכנית לwatchdog כאל הserver ואל better_ping כאל הclient.)
ב watchdog, אנו נמתין לחיבור TCP מהתוכנית ה Better Ping.
לאחר שנוצר חיבור, הwatchdog פשוט ממתין לאות אישור מה-Better Ping. שאכן הגיע אליו תגובת הפינג.
אם הwatchdog לא קיבל **מיד** שום אות, הוא לא יחכה לקבל, הוא מוגדר כMSG_DONTWAIT שזו שיטה של פרוטוקול tcp לעשות את הפעולה שהסברנו של הnon-blocking.
לכן אם הוא לא קיבל אות הוא מיד יגדיל באחד את האינדקס time שמכיל את השניות שעברו וילך לישון שנייה אחת. לאחר מכן יצפה לקבל אות מהbetter_Ping שוב.
הtime סופר כמה שניות בעצם **לא התקבל אות** מהbetter_ping שמייצג שהתקבלה תגובה מהפינג.
בעצם הtime ממש סופר את השניות לקבלת החבילת חזרה של הפינג.
(ומכוון שכל פעם אחרי שמעלים את האינדקס ישנים שנייה זה יוצא בסוף בערך 10 שניות)
אם הוא הwatchdog יקבל לפתע אות אישור '1' ,שזה אומר שהbetter_ping קיבל פינג בחזרה.

הוא מיד יאפס את הטיימר שלו, וילך לישון שנייה אחת.
ויחכה שוב לקבל אות מחדש. על קבלת פינג של השליחה הבאה.
ואם קורה מצב שהאינדקס timer (שמכיל את השניות שעברו) עבר את המספר 10 , הוא ישלח אות '0' לbetter_ping שזה אות שמייצג **נכשל** בתוכנית Better Ping וזה יצא מהלולאת קבלה של better_ping ויגרום לסגירת התוכנית better_ping.

נרצה להסביר שוב יותר בפירוט איך כל זה נראה מהצד של better_ping,
לאחר ש better_ping הפעיל את התוכנית watchdog , התוכנית תנסה להתחבר לתוכנית Watchdog דרך שקע ה-TCP.
לאחר יצירת חיבור (אם הצליח),
התוכנית Better Ping תכין חבילת ECHO REQUEST עם כל הפרמטרים שצריך.
ותשלח את החבילת icmp אל הכתובת IP של היעד.
לאחר מכן, אנחנו נרצה לקבל חבילת ICMP ECHO REPLAY מהIP יעד שלנו.
מכיוון שהsocket RAW נמצא במצב של nonBlocking, אז אם קורה מצב שלא נקבל תגובה מיד , אז התוכנית פשוט תמשיך לפקודה הבאה ולא תחכה לקבלת החבילה.
ואז נבדוק את המצבים השונים:

- אם לא קיבלנו חבילה במיידי, ננסה לקבל שוב את החבילה.
  חשוב להבין, שאם עבר 10 שניות הwatchdog ישלח לנו אות ויעצור אותנו, אבל כל עוד לא עבר 10 שניות, הוא לא שולח לנו אות לעצירה ואנחנו ממשיכים לנסות לקבל.
- אם התקבלה החבילה מיד נצא מהלולאת קבלה ונדפיס את הנתונים.
  לאחר מכן נשלח לwatchdog אות שמראה שאכן התקבלה החבילה ואז יתאפס אצלו הtimer.
  חשוב להבין שסוף סוף הוא יקבל לפונקציית recv שלו את האות אליו הוא חיכה, כל שאר הפעמים הוא פשוט לא יקבל כלום.
  לאחר שליחת האות שולחים פינג נוסף... באותה דרך.
- אם עבר יותר מ10 שניות- שזה קורה בגלל שאנחנו לא עונים (לא שולחים שום אות שאכן התקבל) בכל פעם שאנחנו לא עונים הwatchdog במקביל יספור את מספר הפעמים שלא ענינו.
  וכאשר זה יגיע ל10 הוא ישלח אות סיום-
  הwatchdog ידאג לשלוח אות '0' שזה אומר לbetter_ping לצאת מהתוכנית ולסגור סוקטים.

התוכנית תמשיך לשלוח חבילות עד שהמשתמש יחליט לסגור את Watchdog או עד שהwatchdog יבחין שלקח יותר מידי זמן לקבל את החבילה ויסגור מעצמו את התוכנית.

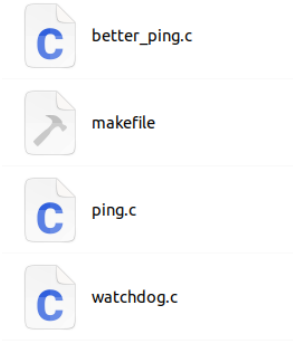
חשוב להבהיר שהמצב non blocking עוזר לכך שבפונקציית recv אנחנו לא נחכה לנצח אלא נצא אחרי זמן מסוים.
וישנה שיטה דומה בtcp שנקראת MSG DONTWAIT .

איך מריצים את התוכנית-

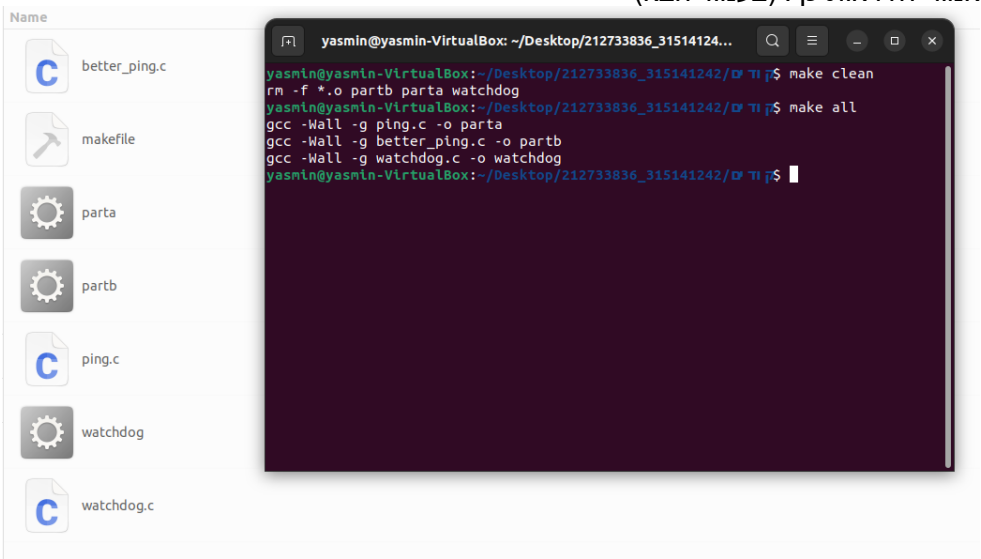
בזיפ שהעלנו יש שתי תיקיות, אחד של הקודים והשני של ההקלטות wireshark,
צריך לחלץ את התקיה הכללית, ובתקיה של הקוד ישנם 3 קבצים להרצה שלבנתיים הם בסיומת c . ובנוסף יש קובץ readme שזה הקובץ הזה.

1. ping.c - תוכנית הפינג הפשוטה - חלק א'.
2. better_ping.c - תוכנית הפינג שמיישמת את ה watchdog - חלק ב'.
3. watchdog.c - תוכנית שמריצים אותה בתוך הbutter_ping.c לא ידנית, רק צריך לקמפל אותה.
4. makefile – משתמשים בו כדי שיהיה נוח לקמפל את התוכנית כולה בבת אחת כדי שיהיה מוכן להרצה.

הנה הקבצים-



צריך דבר ראשון להשתמש בmakefile .
פותחים טרמינל בתקיה שנמצאים בה כל הקבצי c, בתקיית הקודים,
וכותבים בה את הפקודה- *make all*.
לאחר מכן, יווצרו קבצים נוספים, מה שחשובים הם:
parta - קובץ ההפעלה של Ping. c חלק א' במטלה.
partb - קובץ ההפעלה Better Ping. c חלק ב' של המטלה.
Watchdog - קובץ ההפעלה, שאותו רק partb מפעילה מתוכה, ולא אנחנו.
אמור להיראות כך: (בעמוד הבא)



לאחר שיצרנו את הקבצי הרצה.
בשביל להפעיל את חלק א- שזה ping הפשוט, צריך להקליד בטרמינל: *sudo* ./parta "IP"
כאשר "IP" היא כתובת ה-IPv4 היעד שאליו נרצה לשלוח את הפינג.
חשוב להקליד sudo לפני הפקודה, ומכוון שזה פקודה כזו צריך להזין את סיסמת המשתמש .
הIP הדיפולטיבי של התוכנית הוא 8.8.8.8. אם לא נקליד את הIP זה ישלח פינגים לIP של גוגל.
תמונה של הרצה פה למטה...

```
12733836_315141242/ùàâëï$ sudo ./parta 8.8.8.8
[sudo] password for yasmin:
Successfuly sent one packet : ICMP HEADER : 27 bytes, data length : 19 , icmp he
ader : 8
Successfuly received one packet with 47 bytes : data length : 19 , icmp header :
 8 , ip header : 20  seq: 1

RTT: 4.762000 milliseconds (4762 microseconds)
Successfuly sent one packet : ICMP HEADER : 27 bytes, data length : 19 , icmp he
ader : 8
Successfuly received one packet with 47 bytes : data length : 19 , icmp header :
 8 , ip header : 20  seq: 2

RTT: 4.685000 milliseconds (4685 microseconds)
Successfuly sent one packet : ICMP HEADER : 27 bytes, data length : 19 , icmp he
ader : 8
Successfuly received one packet with 47 bytes : data length : 19 , icmp header :
 8 , ip header : 20  seq: 3

RTT: 4.932000 milliseconds (4932 microseconds)
Successfuly sent one packet : ICMP HEADER : 27 bytes, data length : 19 , icmp he
ader : 8
Successfuly received one packet with 47 bytes : data length : 19 , icmp header :
```

כדי להפסיק את הריצה האינסופית הזאת של חלק א' נצטרך ללחוץ על control+c.

```
yasmin@yasmin-VirtualBox: ~/Desktop/final
Successfuly received one packet with 47 bytes : data length : 19 , icmp header : 8 , ip header : 20  seq: 20

RTT: 104.057999 milliseconds (104058 microseconds)
Successfuly sent one packet : ICMP HEADER : 27 bytes, data length : 19 , icmp header : 8
Successfuly received one packet with 47 bytes : data length : 19 , icmp header : 8 , ip header : 20  seq: 21

RTT: 100.427002 milliseconds (100427 microseconds)
Successfuly sent one packet : ICMP HEADER : 27 bytes, data length : 19 , icmp header : 8
Successfuly received one packet with 47 bytes : data length : 19 , icmp header : 8 , ip header : 20  seq: 22

RTT: 102.494003 milliseconds (102494 microseconds)
Successfuly sent one packet : ICMP HEADER : 27 bytes, data length : 19 , icmp header : 8
Successfuly received one packet with 47 bytes : data length : 19 , icmp header : 8 , ip header : 20  seq: 23

RTT: 102.439003 milliseconds (102439 microseconds)
Successfuly sent one packet : ICMP HEADER : 27 bytes, data length : 19 , icmp header : 8
Successfuly received one packet with 47 bytes : data length : 19 , icmp header : 8 , ip header : 20  seq: 24

RTT: 101.903000 milliseconds (101903 microseconds)
Successfuly sent one packet : ICMP HEADER : 27 bytes, data length : 19 , icmp header : 8
Successfuly received one packet with 47 bytes : data length : 19 , icmp header : 8 , ip header : 20  seq: 25

RTT: 101.832001 milliseconds (101832 microseconds)
Successfuly sent one packet : ICMP HEADER : 27 bytes, data length : 19 , icmp header : 8
Successfuly received one packet with 47 bytes : data length : 19 , icmp header : 8 , ip header : 20  seq: 26

RTT: 21.775999 milliseconds (21776 microseconds)
Successfuly sent one packet : ICMP HEADER : 27 bytes, data length : 19 , icmp header : 8
Successfuly received one packet with 47 bytes : data length : 19 , icmp header : 8 , ip header : 20  seq: 27

RTT: 81.028999 milliseconds (81029 microseconds)
Successfuly sent one packet : ICMP HEADER : 27 bytes, data length : 19 , icmp header : 8
^C
yasmin@yasmin-VirtualBox:~/Desktop/final$
```

כדי להפעיל את תוכנית butter_ping נקליד את הפקודה בטרמינל:

sudo ./partb IP

כאשר IP היא כתובת ה-IPv4 של מחשב היעד שנרצה לשלוח אליו את הפינג.

צריך לשים לב שלא צריך לאתחל את תוכנית Watchdog , מכוון שהיא מופעלת לבד על ידי Better Ping.

(אם נריץ לבד זה פשוט לא יעבוד.)

גם פה אם לא מקלידים Ip בפקודת הרצה, אז ה ip הדיפולטיבי הוא 8.8.8.8.

דוגמא להרצה של חלק ב'- פה למטה.. הIP יכול להיות כל דבר לא דווקא 8.8.8.8 כמו בדוגמא.

```
yasmin@yasmin-VirtualBox: ~/Downloads/212733836_315141...
yasmin@yasmin-VirtualBox:~/Downloads/212733836_315141242/212733836_315141242/cod
e$ make all
gcc -Wall -g ping.c -o parta
gcc -Wall -g better_ping.c -o partb
gcc -Wall -g watchdog.c -o watchdog
yasmin@yasmin-VirtualBox:~/Downloads/212733836_315141242/212733836_315141242/cod
e$ sudo ./partb 8.8.8.8
[sudo] password for yasmin:
conecct to IP: 8.8.8.8
Successfuly received one packet  frome IP : 8.8.8.8 received one packet with 47
bytes , seq: 1 ,time: 33.061001 milliseconds and 33061 microseconds
Successfuly received one packet  frome IP : 8.8.8.8 received one packet with 47
bytes , seq: 2 ,time: 28.118000 milliseconds and 28118 microseconds
Successfuly received one packet  frome IP : 8.8.8.8 received one packet with 47
bytes , seq: 3 ,time: 24.931000 milliseconds and 24931 microseconds
Successfuly received one packet  frome IP : 8.8.8.8 received one packet with 47
bytes , seq: 4 ,time: 22.481001 milliseconds and 22481 microseconds
^C
yasmin@yasmin-VirtualBox:~/Downloads/212733836_315141242/212733836_315141242/cod
e$
```

אם החיבור נכשל או לקח יותר מידי זמן לחבילה להגיע watchdog תשלח אות של נכשל לתוכנית Better Ping ותוכנית תסתיים .

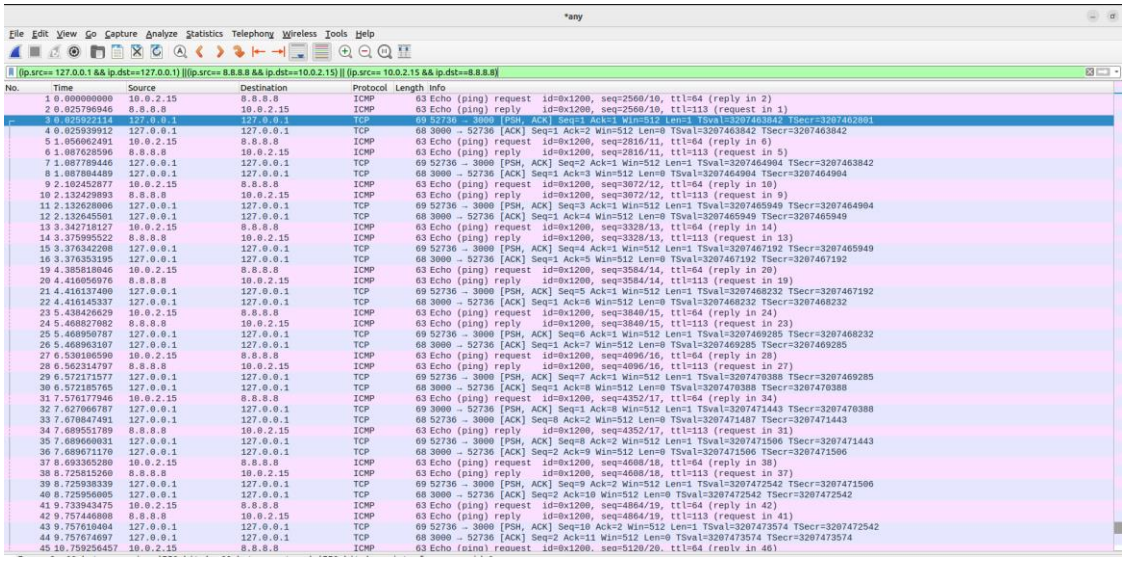
```
yasmin@yasmin-VirtualBox: ~/Downloads/212733836_315141...
yasmin@yasmin-VirtualBox:~/Downloads/212733836_315141242/212733836_315141242/cod
e$ sudo ./partb 8.8.8.127
[sudo] password for yasmin:
conecct to IP: 8.8.8.127
Server 8.8.8.127 cannot be reached.
yasmin@yasmin-VirtualBox:~/Downloads/212733836_315141242/212733836_315141242/cod
e$
```

וכמו שהסברנו לא צריך להריץ את הwatchdog בנפרד זה לא יעבוד.

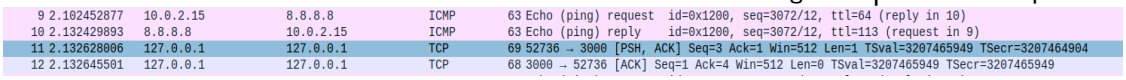
```
yasmin@yasmin-VirtualBox:~/Desktop/final$ sudo ./watchdog
^C
yasmin@yasmin-VirtualBox:~/Desktop/final$
```

-Wireshark

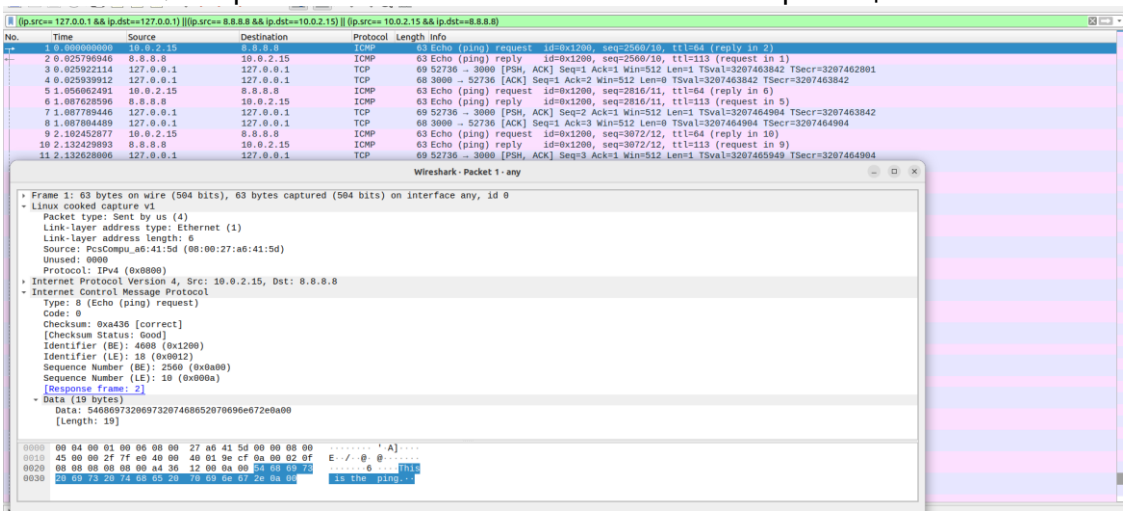
נראה קצת איך התעבורה של התוכניות שכתבנו עוברות ברשת:
צילמנו מסך כאן למטה, כאשר מריצים את ה better_ping:
ניתן לשים לב שבחבילות שכתוב בפרוטוקול ICMP אלו חבילות שמתקבלות כאשר שולחים או מקבלים הודעת ping.
פרוטוקול TCP זה כאשר הwatchdog מתקשר עם הbetter_ping , ניתן לשים לב שבפאקטות האלו גם הsource וגם הdestination הם 127.0.0.1 , כיוון ששני התוכניות מדברות מאותו מחשב.
החבילות שנשלחות בניהם מכילות את אותות אישור מוסכמים בניהם.
ב- destination כאשר כתוב 8.8.8.8 זה בעצם כאשר אנחנו שולחים לגוגל פינג, והמחשב שלנו נמצא בIP- 10.0.2.15.
כאשר זה הפוך (הsrc הופך לdest) זה מחזיר תגובת פינג מגוגל אלינו.
ניתן לראות את חבילות הICMP שהם או request או reply ,אלו חבילות שליחה וקבלה פינג בהתאמה.



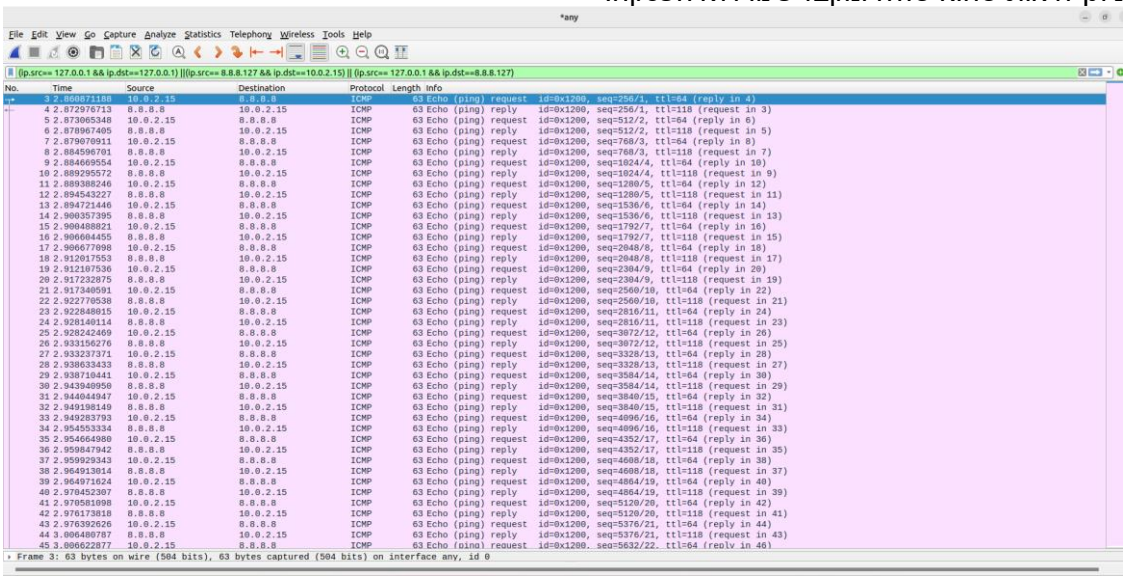
כאן ניתן לראות את פעילות הbetter_ping כאשר הוא מצליח לשלוח ולקבל פינג, הוא שולח ומקבל חבילת icmp ושולח ומקבל מהwatchdog אות אישור.



כאשר פותחים חבילת icmp ניתן לראות את המאפיינים של החבילה לדוגמא מה אורך הודעת הפינג.



נרצה להראות גם איך נראית התעבורה כשמריצים רק את ה- ping של חלק א'.
ניתן לראות שהוא שולח ומקבל פינג ללא הפסקה.



כאשר לא מתקבלת תגובת פינג ב better ping , אז התוכנית watchdog לאחר 10 שניות תסגור את החיבור.
ניתן לראות שיש רק בקשת פינג מipg שגוי (בוורוד) 8.8.8.127 ולכן ולא מתקבלת תגובה ועוברים העשר שניות.
כל העשר שניות האלו הwatchdog בודק האם better ping שלחה לו את האות המוסכם הפינג התקבל, לבסוף מכוון שלא התקבל האות, watchdog שולח לbetter_ping אות סיום שגורמת לסגור את התוכנית.

icmp||tcp

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	127.0.0.1	127.0.0.1	TCP	76	53396 → 3000 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=514420101 TSecr=0 WS=128
2	0.000027388	127.0.0.1	127.0.0.1	TCP	76	3000 → 53396 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=514420101 TSecr=514
3	0.000050303	127.0.0.1	127.0.0.1	TCP	68	53396 → 3000 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=514420101 TSecr=514420101
4	0.000121667	10.0.2.15	8.8.8.127	ICMP	63	Echo (ping) request id=0x1200, seq=256/1, ttl=64 (no response found!)
5	0.000183514	127.0.0.1	127.0.0.1	TCP	69	3000 → 53396 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=1 TSval=514420101 TSecr=514420101
6	0.000196378	127.0.0.1	127.0.0.1	TCP	68	53396 → 3000 [ACK] Seq=1 Ack=2 Win=65536 Len=0 TSval=514420101 TSecr=514420101
7	1.000369535	127.0.0.1	127.0.0.1	TCP	69	3000 → 53396 [PSH, ACK] Seq=2 Ack=1 Win=65536 Len=1 TSval=514421101 TSecr=514420101
8	1.000381053	127.0.0.1	127.0.0.1	TCP	68	53396 → 3000 [ACK] Seq=1 Ack=3 Win=65536 Len=0 TSval=514421101 TSecr=514421101
9	2.000622414	127.0.0.1	127.0.0.1	TCP	69	3000 → 53396 [PSH, ACK] Seq=3 Ack=1 Win=65536 Len=1 TSval=514422102 TSecr=514421101
10	2.000632368	127.0.0.1	127.0.0.1	TCP	68	53396 → 3000 [ACK] Seq=1 Ack=4 Win=65536 Len=0 TSval=514422102 TSecr=514422102
11	3.001605600	127.0.0.1	127.0.0.1	TCP	69	3000 → 53396 [PSH, ACK] Seq=4 Ack=1 Win=65536 Len=1 TSval=514423103 TSecr=514422102
12	3.001623857	127.0.0.1	127.0.0.1	TCP	68	53396 → 3000 [ACK] Seq=1 Ack=5 Win=65536 Len=0 TSval=514423103 TSecr=514423103
13	4.002034053	127.0.0.1	127.0.0.1	TCP	69	3000 → 53396 [PSH, ACK] Seq=5 Ack=1 Win=65536 Len=1 TSval=514424103 TSecr=514423103
14	4.002045650	127.0.0.1	127.0.0.1	TCP	68	53396 → 3000 [ACK] Seq=1 Ack=6 Win=65536 Len=0 TSval=514424103 TSecr=514424103
15	5.002207457	127.0.0.1	127.0.0.1	TCP	69	3000 → 53396 [PSH, ACK] Seq=6 Ack=1 Win=65536 Len=1 TSval=514425103 TSecr=514424103
16	5.002219619	127.0.0.1	127.0.0.1	TCP	68	53396 → 3000 [ACK] Seq=1 Ack=7 Win=65536 Len=0 TSval=514425103 TSecr=514425103
17	6.017840300	127.0.0.1	127.0.0.1	TCP	69	3000 → 53396 [PSH, ACK] Seq=7 Ack=1 Win=65536 Len=1 TSval=514426119 TSecr=514425103
18	6.017850007	127.0.0.1	127.0.0.1	TCP	68	53396 → 3000 [ACK] Seq=1 Ack=8 Win=65536 Len=0 TSval=514426119 TSecr=514426119
19	7.017996726	127.0.0.1	127.0.0.1	TCP	69	3000 → 53396 [PSH, ACK] Seq=8 Ack=1 Win=65536 Len=1 TSval=514427119 TSecr=514426119
20	7.018008272	127.0.0.1	127.0.0.1	TCP	68	53396 → 3000 [ACK] Seq=1 Ack=9 Win=65536 Len=0 TSval=514427119 TSecr=514427119
21	8.021082116	127.0.0.1	127.0.0.1	TCP	69	3000 → 53396 [PSH, ACK] Seq=9 Ack=1 Win=65536 Len=1 TSval=514428122 TSecr=514427119
22	8.021091797	127.0.0.1	127.0.0.1	TCP	68	53396 → 3000 [ACK] Seq=1 Ack=10 Win=65536 Len=0 TSval=514428122 TSecr=514428122
23	9.028359961	127.0.0.1	127.0.0.1	TCP	69	3000 → 53396 [PSH, ACK] Seq=10 Ack=1 Win=65536 Len=1 TSval=514429129 TSecr=514428122
24	9.028372167	127.0.0.1	127.0.0.1	TCP	68	53396 → 3000 [ACK] Seq=1 Ack=11 Win=65536 Len=0 TSval=514429129 TSecr=514429129
25	10.028516987	127.0.0.1	127.0.0.1	TCP	69	3000 → 53396 [PSH, ACK] Seq=11 Ack=1 Win=65536 Len=1 TSval=514430130 TSecr=514429129
26	10.028556848	127.0.0.1	127.0.0.1	TCP	68	53396 → 3000 [ACK] Seq=1 Ack=12 Win=65536 Len=0 TSval=514430130 TSecr=514430130
27	10.028580404	127.0.0.1	127.0.0.1	TCP	68	3000 → 53396 [FIN, ACK] Seq=12 Ack=1 Win=65536 Len=0 TSval=514430130 TSecr=514430130
28	10.028700571	127.0.0.1	127.0.0.1	TCP	68	53396 → 3000 [FIN, ACK] Seq=1 Ack=13 Win=65536 Len=0 TSval=514430130 TSecr=514430130
29	10.028704913	127.0.0.1	127.0.0.1	TCP	68	3000 → 53396 [ACK] Seq=13 Ack=2 Win=65536 Len=0 TSval=514430130 TSecr=514430130

כאשר אנחנו עוצרים את better ping באמצע, (על ידי control+C) אנחנו בעצם יכולים לפגוע בפאקטות שבעת העצירה היו באמצע להישלח. ולכן הפקטה האחרונה פה צבועה באדום כי היא נפגעה מהסגירה הפתאומית.

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

icmp||tcp

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	127.0.0.1	127.0.0.1	TCP	76	56988 → 3000 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=514910048 TSecr=0 WS=128
2	0.000022824	127.0.0.1	127.0.0.1	TCP	76	3000 → 56988 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=514910048 TSecr=514910048 WS=128
3	0.000044967	127.0.0.1	127.0.0.1	TCP	68	56988 → 3000 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=514910048 TSecr=514910048
4	0.000099849	10.0.2.15	8.8.8.8	ICMP	63	Echo (ping) request id=0x1200, seq=256/1, ttl=64 (reply in 7)
5	0.000519326	127.0.0.1	127.0.0.1	TCP	69	3000 → 56988 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=1 TSval=514910049 TSecr=514910048
6	0.000534697	127.0.0.1	127.0.0.1	TCP	68	56988 → 3000 [ACK] Seq=1 Ack=2 Win=65536 Len=0 TSval=514910049 TSecr=514910049
7	0.012579319	8.8.8.8	10.0.2.15	ICMP	63	Echo (ping) reply id=0x1200, seq=256/1, ttl=118 (request in 4)
8	0.012827939	127.0.0.1	127.0.0.1	TCP	69	56988 → 3000 [PSH, ACK] Seq=1 Ack=2 Win=65536 Len=1 TSval=514910061 TSecr=514910049
9	0.012843963	127.0.0.1	127.0.0.1	TCP	68	3000 → 56988 [ACK] Seq=2 Ack=2 Win=65536 Len=0 TSval=514910061 TSecr=514910061
10	1.016262336	10.0.2.15	8.8.8.8	ICMP	63	Echo (ping) request id=0x1200, seq=512/2, ttl=64 (reply in 11)
11	1.028632302	8.8.8.8	10.0.2.15	ICMP	63	Echo (ping) reply id=0x1200, seq=512/2, ttl=118 (request in 10)
12	1.028807087	127.0.0.1	127.0.0.1	TCP	69	56988 → 3000 [PSH, ACK] Seq=2 Ack=2 Win=65536 Len=1 TSval=514911077 TSecr=514910061
13	1.028826946	127.0.0.1	127.0.0.1	TCP	68	3000 → 56988 [ACK] Seq=2 Ack=3 Win=65536 Len=0 TSval=514911077 TSecr=514911077
14	2.040177821	10.0.2.15	8.8.8.8	ICMP	63	Echo (ping) request id=0x1200, seq=768/3, ttl=64 (reply in 15)
15	2.058702285	8.8.8.8	10.0.2.15	ICMP	63	Echo (ping) reply id=0x1200, seq=768/3, ttl=118 (request in 14)
16	2.058792538	127.0.0.1	127.0.0.1	TCP	69	56988 → 3000 [PSH, ACK] Seq=3 Ack=2 Win=65536 Len=1 TSval=514912107 TSecr=514911077
17	2.058808739	127.0.0.1	127.0.0.1	TCP	68	3000 → 56988 [ACK] Seq=2 Ack=4 Win=65536 Len=0 TSval=514912107 TSecr=514912107
18	2.538074072	127.0.0.1	127.0.0.1	TCP	68	3000 → 56988 [RST, ACK] Seq=2 Ack=4 Win=65536 Len=0 TSval=514912587 TSecr=514912107

סוף.