מסמך pdf מטלה 4 ברשתות תקשורת

ראשית, נסביר קצת על מטרת המטלה והסבר למשתמש איך להריץ.

חלק א'

מטרת המטלה היא לעקוב אחר שליחת פינגים (ping). כדי לשלוח ping ראשית נפתח Raw_Socket, ע"י פקודה, (נסביר בהמשך בתמונה מהקוד). בחלק א' של המטלה, אנחנו רק נעקוב אחר שליחת פינגים ל פורט 8.8.8.8 – הפורט של גוגל, נרצה לשלוח פינגים ללא הפסקה, כלומר במקרה שנרצה לעצור את התוכנית נפסיק ידנית ע"י Ctrl+c. בנוסף, מה שנרצה לראות בחלק א' של המטלה, זה שבאמת התקבל ping וקיבלנו תגובה חזרה(נראה זאת ב הקלטות וצילומי מסך של ה wireshark) , כלומר, נרצה לשלוח ICMP ECHO REQUEST ונרצה לקבל תשובה חזרה, כלומר נרצה לראות בתגובה ICMP ECHO REPLY וברגע שנקבל את התגובה, שוב פעם נשלח פינגים עד שהמשתמש יפסיק ידנית את התוכנית.

חלק ב'

בחלק ב' מה שנוסיף על חלק א' זה איזשהו טיימר, שימדוד את הזמן מרגע השליחה של ה ping עד לתגובה על השליחה, ובמקרה שהזמן שמדד גבוה מ-10 שניות, אז התוכנית תיסגר, ה watchdog יסגור את התוכנית ע"י הפקודה kill שהיא תהרוג את ה process של האבא, כלומר את better_ping, ולאחר מכן יסגור את ה socket שיצרנו ל Tcp connection.

איך נעשה זאת?

ניצור חיבור Tcp בין ה watchdog לבין ה better_ping וברגע שנשלח ping אז נשלח הודעה ל watchdog שיתחיל את הטיימר וברגע שנקבל תגובה על ה ping נשלח הודעה ל watchdog שיאפס את הטיימר.

במטלה הגשנו את קבצי הקוד של watchdog.c better_ping.c ping.c ו makefile, ה makefile יקמפל את הקבצים בצורה הבאה. את ping.c יקמפל לפקודה parta כלומר כדי להריץ את חלק א' של המטלה נריץ את הפקודה parta/. אבל פקודה זו לבד לא מספיקה, אם נריץ את הקוד על ה Ubunto נריץ את הפקודה עם תוספת של הרשאה למנהל, כלומר sudo ונוסיף גם את ה IP שנרצה לעשות לו ping לדוגמא: sudo ./parta 8.8.8.8 .

את חלק ב' של המטלה נריץ בצורה הבאה. נקמפל את watchdog.c כרגיל לפקודה watchdog, ואת better_ping נקמפל לפקודה partb/., ונסביר מה קורה בהרצה. בתחילת הקוד של ה better_ping שמנו כמה שורות קוד, כך שיעשה fork ל watchdog , נשים לב שבמערך שהבאנו לו ב fork שמנו לו את הפקודת הרצה של ה watchdog ולכן, ברגע שנריץ את ה better_ping אז הוא

יריץ גם את ה watchdog ואז יפתח קשר Tcp ביניהם והקוד ימשיך כמו שהסברנו למעלה.

מצ"ב תמונה שמה שהסברנו איך פועלת ההרצה של חלק ב.

```
// run 2 programs using fork + exec
// command: make clean && make all && ./partb
int main(int count, char *argv[])
{
    char *args[2];
    // compiled watchdog.c by makefile
    args[0] = "./watchdog";
    args[1] = NULL;
    int status;
    int pid = fork();
    if (pid == 0)
    {
        printf("in child \n");
        execvp(args[0], args);
    }
    printf("child exit status is: %d\n", status);
```

לאחר שהסברנו על מטרת המטלה ואיך המשתמש יריץ אותה, נסביר קצת על הפונקציות שיש בקוד.

ראשית קובץ ה ping של חלק א (נצרף תמונות לאחר ההסבר)

בתחילת הקוד נרצה להצהיר על הפונקציה של ה checksum שנשתמש בה בהמשך, לאחר מכן נפתח raw_socket, ונשלח ping, לאחר ששלחנו ping נעשה לולאת קבלה, לקבל תגובה ל pingששלחנו, לאחר מכן, אם קיבלנו תגובה כמו שצריך, אז נצא מהלולאה של הקבלה ונחזור לתחילת הלולאה של while(1) כלומר, נשלח שוב פעם פינגים, נגדיר מחדש את מה שצריך במקרה וצריך, ונחכה לתגובה ל ping וכן הלאה..

ב קובץ של better_ping

ראשית, נרצה לעשות fork ל watchdog (הראנו צילום מסך והסבר למעלה), לאחר מכן נאשר את חיבור ה Tcp עם ה watchdog , ונפתח raw_socket בכדי לשלוח ping – בדומה למה שעשינו בחלק א, לאחר שפתחנו raw_socket נרצה לשלוח הודעה ל watchdog שיתחיל את הטיימר, לאחר ששלחנו את ההודעה נכנס ללואת while(1) – לולאה אינסופית, שבה נגדיר מה שצריך בתחילת הלולאה, לאחר מכן נשלח ping ל IP שנרצה, ואז ניכנס ללולאת קבלה בדומה לחלק א', ובלולאת קבלה, לאחר שנקבל Reply נרצה לשלוח הודעה ל watchdog שיאפס שוב פעם את הטיימר, ולאחר מכן נצא מהלולאת קבלה

ונחזור לתחילת הלולאה האינסופית, כלומר נשלח פינג שוב פעם, נקבל תגובה לפינג, ונשלח הודעה ל watchdog שיאפס את הטיימר.

וכך התוכנית תמשיך, התוכנית תעצור באחד משני מקרים:

מקרה ראשון, שהזמן שעבר בין שליחת הפינג עד לתגובה על הפינג הוא לפחות 10 שניות, במקרה זה, ה טיימר יקלוט זאת ויסגור את התוכנית.

מקרה שני, במקרה שהמשתמש יחליט להפסיק את התוכנית ידנית.

מצ"ב קטעי הקוד שהסברנו לעיל:

```
// Create raw socket for IP-RAW (make IP-header by yourself)
int sock = -1;
if ((sock = socket(AF_INET, SOCK_RAW, IPPROTO_ICMP)) == -1)
{
    fprintf(stderr, "socket() failed with error: %d\n", errno);
    fprintf(stderr, "To create a raw socket, the process needs to be run by Admin/root user.\n\n");
    return -1;
}

//Set the IP address of the host to ping
struct sockaddr_in dest_in;
memset(&dest_in, 0, sizeof(struct sockaddr_in));
dest_in.sin_family = AF_INET;
dest_in.sin_addr.s_addr = inet_addr(DesIp);

//Create time struct
struct timeval start, end;
int sequence = 0;
```

```
    //Receiving loop
    while ((bytes_received = recvfrom(sock, packet, sizeof(packet), 0, (struct sockaddr *)&dest_in, &len)))
    {
        // Extract the IP header and ICMP header from the received packet
        char *ip_header = packet;
        int ip_header_len = (ip_header[0] & 0x0f) * 4;
        struct icmphdr *icmp_reply = (struct icmphdr*)(packet + ip_header_len);
        if (bytes_received > 0)
        {
            gettimeofday(&end, 0);
            float milliseconds = (end.tv_sec - start.tv_sec) * 1000.0f + (end.tv_usec - start.tv_usec) / 1000.0f;
            unsigned long microseconds = (end.tv_sec - start.tv_sec) * 1000.0f + (end.tv_usec - start.tv_usec);
            // Check the IP header
            struct iphdr *iphdr = (struct iphdr *)packet;
            struct icmphdr *icmphdr = (struct icmphdr *)(packet + (iphdr->ihl * 4));


            printf("%s: seq=%d time=%fms\n", inet_ntoa(dest_in.sin_addr), icmp_reply->un.echo.sequence, milliseconds);

            break;
        }
    }
```

שתי תמונות אלה הם מקובץ ה ping, התמונה הראשונה היא יצירת ה raw_socket והתמונה השנייה היא לולאת הקבלה כתגובה לשליחת הפינג.

```c
//Create time struct
struct timeval start, end;
int sequence = 0;

int checkSend = -1;
    while (checkSend < 0 )
{
    int pingSend = 300;
    checkSend = send(sockfd, &pingSend , sizeof(pingSend),0);
    if(checkSend > 0)
    {
        printf("Sent start to watchdog\n");
    }
    else{
        printf("Error in send\n");
    }
}
```

```c
//Receiving loop
while ((bytes_received = recvfrom(sock, packet, sizeof(packet), 0, (struct sockaddr *)&dest_in, &len)))
{
    // Extract the IP header and ICMP header from the received packet
    char *ip_header = packet;
    int ip_header_len = (ip_header[0] & 0x0f) * 4;
    struct icmphdr *icmp_reply = (struct icmphdr*)(packet + ip_header_len);
    if (bytes_received > 0)
    {
        gettimeofday(&end, 0);
        float milliseconds = (end.tv_sec - start.tv_sec) * 1000.0f + (end.tv_usec - start.tv_usec) / 1000.0f;
        unsigned long microseconds = (end.tv_sec - start.tv_sec) * 1000.0f + (end.tv_usec - start.tv_usec);
        // Check the IP header
        struct iphdr *iphdr = (struct iphdr *)packet;
        struct icmphdr *icmphdr = (struct icmphdr *)(packet + (iphdr->ihl * 4));


        printf("%s: seq=%d time=%fms\n", inet_ntoa(dest_in.sin_addr), icmp_reply->un.echo.sequence, milliseconds);

        int pingSend = 256;
        int checkSend = send(sockfd, &pingSend , sizeof(pingSend),0);
        if(checkSend > 0)
        {
            printf("watchdog, make the timer zero!!\n");
        }
        else{
            printf("Error in send\n");
        }

        break;
    }
}
```

שתי תמונות אלה הם מקובץ ה better_ping, בתמונה הראשונה אנו שולחים הודעה ל watchdog שידע שאנחנו הולכים לשלוח פינג ושיתחיל להאזין, בתמונה השנייה יש את לולאת הקבלה כתגובה לשליחת הפינג, ובנוסף לאחר קבלת התגובה נשלח הודעה ל watchdog שיאפס את הטיימר כי קיבלנו תגובה.

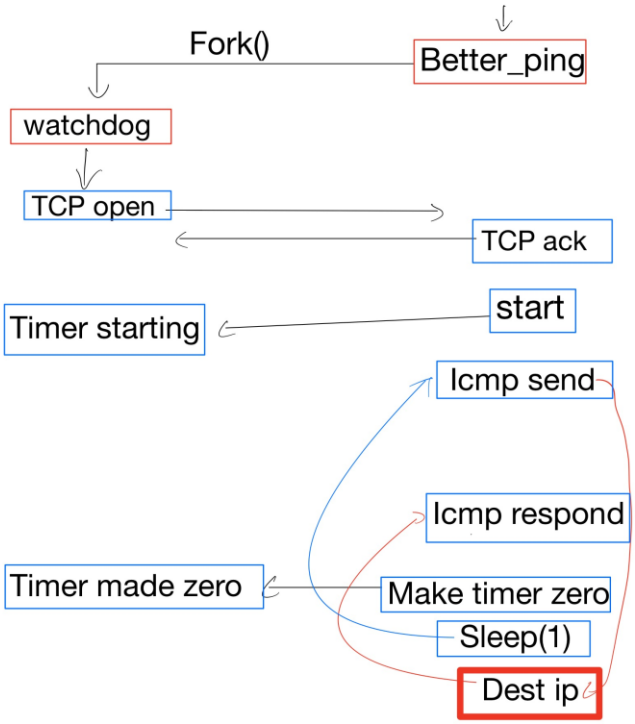
```c
int recvPing = 0;

while (recvPing != 300)
{
    int return_status = recv(clientSocket, &recvPing, sizeof(recvPing),MSG_DONTWAIT);
    if (return_status <= 0) {
        printf("Problem in receiving start\n");
    }
    if(recvPing == 300)
    {
        printf("start mesg recievde\n");      You, now • Uncommitted changes
    }
}
recvPing = 0;
```

```c
    fcntl(listeningSocket, F_SETFL, O_NONBLOCK);
    fcntl(clientSocket, F_SETFL, O_NONBLOCK);



    //Create time variables for timer
    struct timeval tv;
    tv.tv_sec = 10;
    tv.tv_usec = 0;      You, 1 minute ago • Uncommitted changes

    setsockopt(clientSocket, SOL_SOCKET, SO_RCVTIMEO, (const char *)&tv, sizeof tv);


    while(1)
    {
        sleep(3);
        int return_status = recv(clientSocket, &recvPing, sizeof(recvPing),0);
        if (return_status <= 0) {
            printf("Problem in receiving \n");
            break;
        }
        recvPing = 0;
    }

    kill(0,SIGINT); // kill all processes in the process group
    close(clientSocket);
    close(listeningSocket);

    return 0;
}
```

שתי תמונות אלה הם מקובץ ה watchdog.

בתמונה הראשונה אנחנו מקבלים את ההודעה שלפני הלולאת while שלהיות מוכן שהוא שולח פינג ולהתחיל להאזין, לכן, לאחר מכן נהפוך את הסוקט למצב Non-Blocking ונגדיר משתנים בשביל הזמן, ונגדיר לסוקט Timeout של 10. לאחר מכן ניכנס ללולאה אינסופית של קבלה, מה שיהיה בלולאה זה קבלה, והתהליך יקרה כך, אם לא נקבל פאקטה במשך 10 שניות אז המשתנה return_status לא יקבל כלום, ולכן יכנס לתנאי ויעשה לו break ויסגור את התוכנית.

בנוסף, נצרף תרשים זרימה לחלק ב' של המטלה, במקרה שמשהו לא היה מוסבר כראוי.


Fork()
Better_ping
watchdog
TCP open
TCP ack
start
Timer starting
Icmp send
Icmp respond
Timer made zero
Make timer zero
Sleep(1)
Dest ip

בס״ד

נראה שימוש בתוכית ואת הסנפת הפקטות במהלך השימוש, בדרך זאת נוכל לקבל תמונה רחבה יותר של פעילות המערכת.

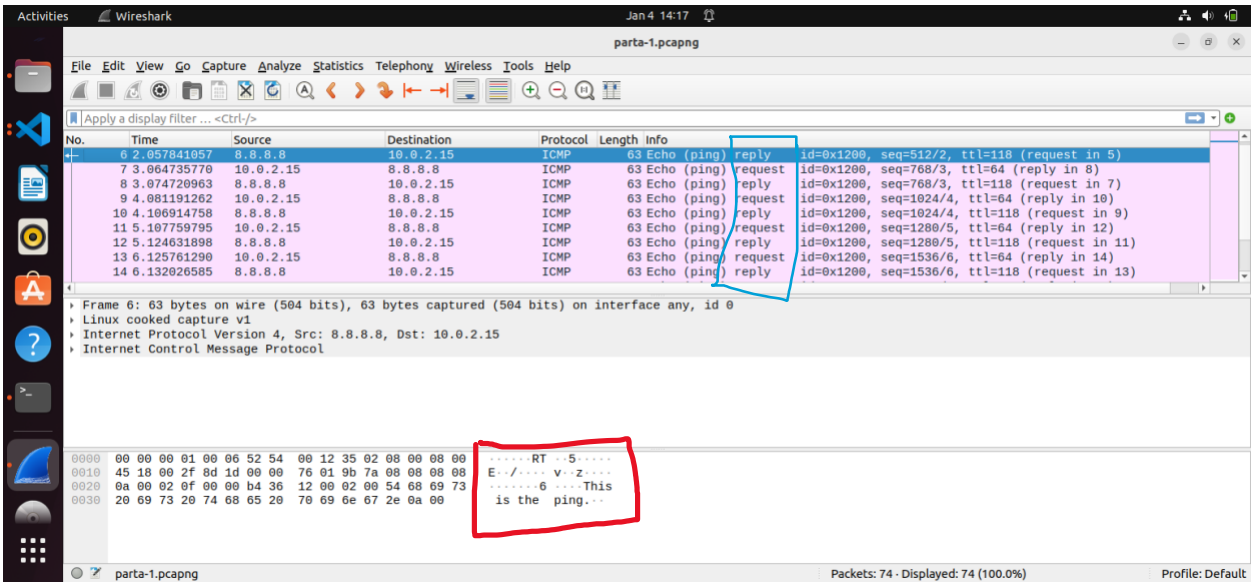
חלק א.
הרצת התוכנית בטרמינל לאחר בניית הקבצים, שימוש בפקודה parta עם נתינת ip רצוי לביצוע הפינג.

```
gcc new_ping.c -o partb
yehonatan@yehonatan-VirtualBox:~/Downloads/4 קבצי עזר לתוכלה/fork + exec$ sudo ./parta 8.8.8.8
[sudo] password for yehonatan:
String: 8.8.8.8
8.8.8.8: seq=0 time=13.355000ms
8.8.8.8: seq=1 time=13.397000ms
8.8.8.8: seq=2 time=7.777000ms
8.8.8.8: seq=3 time=10.051000ms
8.8.8.8: seq=4 time=25.802000ms
8.8.8.8: seq=5 time=16.941000ms
8.8.8.8: seq=6 time=6.327000ms
8.8.8.8: seq=7 time=6.610000ms
8.8.8.8: seq=8 time=15.377000ms
8.8.8.8: seq=9 time=6.226000ms
8.8.8.8: seq=10 time=6.180000ms
8.8.8.8: seq=11 time=5.817000ms
8.8.8.8: seq=12 time=23.735001ms
8.8.8.8: seq=13 time=14.697000ms
8.8.8.8: seq=14 time=8.248000ms
8.8.8.8: seq=15 time=6.598000ms
8.8.8.8: seq=16 time=11.898000ms
8.8.8.8: seq=17 time=7.066000ms
8.8.8.8: seq=18 time=13.607000ms
8.8.8.8: seq=19 time=15.479000ms
8.8.8.8: seq=20 time=20.134001ms
8.8.8.8: seq=21 time=6.496000ms
8.8.8.8: seq=22 time=10.929000ms
8.8.8.8: seq=23 time=13.898000ms
8.8.8.8: seq=24 time=8.447000ms
8.8.8.8: seq=25 time=13.394000ms
8.8.8.8: seq=26 time=13.405000ms
8.8.8.8: seq=27 time=16.490000ms
8.8.8.8: seq=28 time=6.402000ms
8.8.8.8: seq=29 time=16.805000ms
8.8.8.8: seq=30 time=6.399000ms
```

בתמונה ניתן לראות כי לאחר הרצת המערכת היא שולחת פינגים ומדפסיה למשתמש את ה ip שאליו נשלח הפינג את המספר הסידורי של הפינג ברצף השליחות ואת הזמן שחלף בין בקשת הפינג לתגובת השרת שאליו נשלחה הבקשה.

בהסנפת הפקטות התהליך נראה כך:

בס"ד



בצילום מסך זה ניתן לראות בריבוע את המסומן את ההודעה הנשלחת בתוך פקטת ה icmp וכן בחלק העליון את תהליך שליחת הפקטות וקבלת התגובה מהשרת שאליו נשלחה הבקשה.
בריבוע התכלת ניתן לראות את סוג הפקטה, האם זאת פקטת בקשה מהשרת או פקטת תגובה.

<u>חלק ב' better_ping & watchdog.</u>

המערכת השנייה שיצרנו ומופעלת באמצעות הפקודה partb בצירוף ip שאליו אנחנו רוצים לשלוח את הפינג שלנו, מריצה באמצעות הפקודה ()fork את הבן שלה שהוא הטיימר watchdog שיצרנו והסברנו עליו לעיל.
נראה את הטרמינל בעת ההרצה.

בס"ד

```
yehonatan@yehonatan-VirtualBox:~/Downloads/4 עזר למטלה קבצי/fork + exec$ sudo ./partb 8.8.8.8
child exit status is: 0
in child
Hello im watchdog
server is binding

connected to server
Sent start to watchdog
start mesg recievde
8.8.8.8: seq=0 time=6.829000ms
watchdog, make the timer zero!!
8.8.8.8: seq=1 time=6.390000ms
watchdog, make the timer zero!!
8.8.8.8: seq=2 time=13.909000ms
watchdog, make the timer zero!!
8.8.8.8: seq=3 time=8.726000ms
watchdog, make the timer zero!!
8.8.8.8: seq=4 time=5.583000ms
watchdog, make the timer zero!!
8.8.8.8: seq=5 time=9.832000ms
watchdog, make the timer zero!!
8.8.8.8: seq=6 time=6.215000ms
watchdog, make the timer zero!!
8.8.8.8: seq=7 time=5.545000ms
watchdog, make the timer zero!!
8.8.8.8: seq=8 time=6.097000ms
watchdog, make the timer zero!!
8.8.8.8: seq=9 time=6.259000ms
watchdog, make the timer zero!!
8.8.8.8: seq=10 time=6.518000ms
watchdog, make the timer zero!!
8.8.8.8: seq=11 time=6.385000ms
```

ההדפסה הראשונה מקושרת לתהליך ה ()fork לאחר ש watchdog מופעל הגדרנו לו להדפיס למשתמש הודעת הכירות "Hello im watchdog" לאחר מכן נראה את הדפסה של better_ping על כך שהופעל הסרבר והודעת ההתחברות של ה watchdog אליו. לאחר שההתחברות הופעלה בהצלחה better_ping יתן חיווי ל watchdog על כך שהוא מתחיל לשגר פינגים וה watchdog ידפיס שההודעה התקבלה. לאחר מכן אנחנו רואים את ההדפסה של הפינגים שהתקבלו ממש כמו בחלק א'. ולאחר כל קבלה של פינג better_ping שולח הודעת איפוס לטיימר שמודפסת בתור "watchdog, make the timer zero!!" ולאחר מכן נראה את ההדפסות של השליחות הבאות.

נראה כעת את הספקת הפקטות במהלך ריצת החלק השני של המערכת.
צילום המסך הבא מראה את פתיחת הקשר בין better_ping לבין watchdog וזאת בתקשורת הפנימית של המכונה שעליו הורצה המערכת בפרוטוקול tcp.

	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	127.0.0.1	127.0.0.1	TCP	74	48962 → 3000 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=66404973
2	0.000016067	127.0.0.1	127.0.0.1	TCP	74	3000 → 48962 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSv
3	0.000029874	127.0.0.1	127.0.0.1	TCP	66	48962 → 3000 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=664049737 TSecr=66404973

בצילום הבא נראה את שליחת בקשות האיפוס מ better_ping ל watchdog.

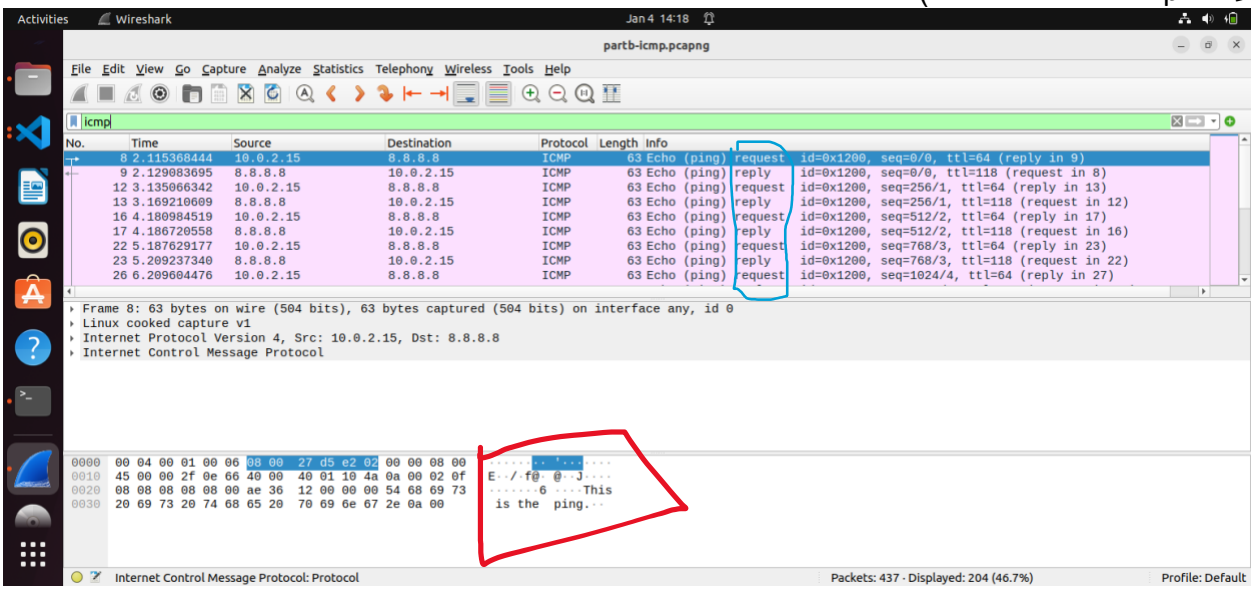
7	0.007030795	127.0.0.1	127.0.0.1	TCP	66	3000 → 48962 [ACK] Seq=1 Ack=9 Win=65536 Len=0 TSval=664049744 TSecr=664049744
8	1.014546752	127.0.0.1	127.0.0.1	TCP	70	48962 → 3000 [PSH, ACK] Seq=9 Ack=1 Win=65536 Len=4 TSval=664050752 TSecr=66404…
9	1.014563990	127.0.0.1	127.0.0.1	TCP	66	3000 → 48962 [ACK] Seq=1 Ack=13 Win=65536 Len=0 TSval=664050752 TSecr=664050752
10	2.029412965	127.0.0.1	127.0.0.1	TCP	70	48962 → 3000 [PSH, ACK] Seq=13 Ack=1 Win=65536 Len=4 TSval=664051767 TSecr=6640…
11	2.029422322	127.0.0.1	127.0.0.1	TCP	66	3000 → 48962 [ACK] Seq=1 Ack=17 Win=65536 Len=0 TSval=664051767 TSecr=664051767
12	3.041686520	127.0.0.1	127.0.0.1	TCP	70	48962 → 3000 [PSH, ACK] Seq=17 Ack=1 Win=65536 Len=4 TSval=664052779 TSecr=6640…
13	3.041695252	127.0.0.1	127.0.0.1	TCP	66	3000 → 48962 [ACK] Seq=1 Ack=21 Win=65536 Len=0 TSval=664052779 TSecr=664052779
14	4.065777307	127.0.0.1	127.0.0.1	TCP	70	48962 → 3000 [PSH, ACK] Seq=21 Ack=1 Win=65536 Len=4 TSval=664053803 TSecr=6640…

בצילום הבא נראה את סגירת הקשר.

205	99.513340822	127.0.0.1	127.0.0.1	TCP	66	3000 → 48962 [ACK] Seq=1 Ack=401 Win=65536 Len=0 TSval=664149251 TSecr=664149251
206	100.535139729	127.0.0.1	127.0.0.1	TCP	70	48962 → 3000 [PSH, ACK] Seq=401 Ack=1 Win=65536 Len=4 TSval=664150272 TSecr=664…
207	100.535150008	127.0.0.1	127.0.0.1	TCP	66	3000 → 48962 [ACK] Seq=1 Ack=405 Win=65536 Len=0 TSval=664150272 TSecr=664150272
208	101.543057705	127.0.0.1	127.0.0.1	TCP	70	48962 → 3000 [PSH, ACK] Seq=405 Ack=1 Win=65536 Len=4 TSval=664151280 TSecr=664…
209	101.543064241	127.0.0.1	127.0.0.1	TCP	66	3000 → 48962 [ACK] Seq=1 Ack=409 Win=65536 Len=0 TSval=664151280 TSecr=664151280
210	102.569790604	127.0.0.1	127.0.0.1	TCP	70	48962 → 3000 [PSH, ACK] Seq=409 Ack=1 Win=65536 Len=4 TSval=664152307 TSecr=664…
211	102.569797721	127.0.0.1	127.0.0.1	TCP	66	3000 → 48962 [ACK] Seq=1 Ack=413 Win=65536 Len=0 TSval=664152307 TSecr=664152307
212	103.561422218	127.0.0.1	127.0.0.1	TCP	66	48962 → 3000 [FIN, ACK] Seq=413 Ack=1 Win=65536 Len=0 TSval=664153299 TSecr=664…

בס"ד

כעת נראה את התקשורת של better_ping עם השרת (במקרה זה השרת של גוגל שכן הורצה הפקודה עם ה ip של השרת שלהם.)



בצילום מסך זה ניתן לראות בריבוע את המסומן את ההודעה הנשלחת בתוך פקטת ה icmp וכן בחלק העליון את תהליך שליחת הפקטות וקבלת התגובה מהשרת שאליו נשלחה הבקשה.
בריבוע התכלת ניתן לראות את סוג הפקטה, האם זאת פקטת בקשה מהשרת או פקטת תגובה.

נראה את תגובת המערכת במקרה קצה שבו better_ping לא מקבל תגובה מהשרת לזמן ארוך יותר מאשר הוגדר לטיימר ב watchdog, ע"מ לבצע זאת הארכנו את פקודת sleep() מהשהייה של שנייה בין קבלת התגובה של הפינג לבין השליחה הבאה של הבקשה להשהייה של 11 שניות מה שגרם לטיימר לקפוץ שכן הוא מאותחל להמתין עשר שניות בלבד.
בצילום המסך של הטרמינל ניתן לראות כי מתקיימים כלל השלבים שהראנו בתחילת סעיף זה ולאחר שהתקבלה תגובה לפינג הראשון ונשלחה בקשת איפוס ראשונה מדפיס לנו ה watchdog כי יש בעיה בקבלת פקטות איפוס הזמן ומבצע סגירה של תקשורת ה tcp בינהם והרצת המערכת מופסקת.

```
hagay@ubuntuM1:~/Desktop/networks_4$ sudo ./partb 8.8.8.8
[sudo] password for hagay:
child exit status is: 0
in child
Hello im watchdog
server is binding

connected to server
Sent start to watchdog
start mesg recievde
8.8.8.8: seq=0 time=6.265000ms
watchdog, make the timer zero!!
Problem in receiving

hagay@ubuntuM1:~/Desktop/networks_4$
```

בס״ד

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	127.0.0.1	127.0.0.1	TCP	74	53430 → 3000 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=2611701218 TSec…
2	0.000039918	127.0.0.1	127.0.0.1	TCP	74	3000 → 53430 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=2611…
3	0.000060336	127.0.0.1	127.0.0.1	TCP	66	53430 → 3000 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=2611701218 TSecr=2611701218
4	0.002195833	127.0.0.1	127.0.0.1	TCP	70	53430 → 3000 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=4 TSval=2611701220 TSecr=2611701218
5	0.009120720	127.0.0.1	127.0.0.1	TCP	66	3000 → 53430 [ACK] Seq=1 Ack=5 Win=65536 Len=0 TSval=2611701227 TSecr=2611701220
6	0.010533190	127.0.0.1	127.0.0.1	TCP	70	53430 → 3000 [PSH, ACK] Seq=5 Ack=1 Win=65536 Len=4 TSval=2611701228 TSecr=2611701227
7	0.010537482	127.0.0.1	127.0.0.1	TCP	66	3000 → 53430 [ACK] Seq=1 Ack=9 Win=65536 Len=0 TSval=2611701228 TSecr=2611701228
8	6.014767715	127.0.0.1	127.0.0.1	TCP	66	3000 → 53430 [FIN, ACK] Seq=1 Ack=9 Win=65536 Len=0 TSval=2611707233 TSecr=2611701228
9	6.014814341	127.0.0.1	127.0.0.1	TCP	66	53430 → 3000 [FIN, ACK] Seq=9 Ack=2 Win=65536 Len=0 TSval=2611707233 TSecr=2611707233
10	6.014825883	127.0.0.1	127.0.0.1	TCP	66	3000 → 53430 [ACK] Seq=2 Ack=10 Win=65536 Len=0 TSval=2611707233 TSecr=2611707233

ניתן לראות בווירשארק כי נשלחות רק בקשת ההתחלה ובקשת איפוס אחת ולאחר מכן נסגר הקשר ולא מתבצעות עוד שליחות שכן קפץ הטיימר.