

מטלה 4 - רשתות תקשורת

מגישים:

שחר זידל - 211990700

מוהנד ספי - 208113381

Ping is a network utility that refers to the signal sent out across the network to another computer, which then sends its own signal back. This signal, which is measured in milliseconds (ms), lets you know how long it takes for a packet of data to travel from your computer to a server on the internet and back

Explain for some functions from the code:-

better_ping.c:-

The [Checksum](#) is an error detection method that detected errors in data/message while it is transmitted from sender to receiver. This method is used by the higher layer protocols and makes use of the Checksum Generator on the Sender side and Checksum Checker on the Receiver side.

```
// Compute checksum (RFC 1071).
unsigned short calculate_checksum(unsigned short *paddress, int len)
{
    int nleft = len;
    int sum = 0;
    unsigned short *w = paddress;
    unsigned short answer = 0;

    while (nleft > 1)
    {
        sum += *w++;
        nleft -= 2;
    }

    if (nleft == 1)
    {
        *((unsigned char *)&answer) = *((unsigned char *)w);
        sum += answer;
    }

    // add back carry outs from top 16 bits to low 16 bits
    sum = (sum >> 16) + (sum & 0xffff); // add hi 16 to low 16
    sum += (sum >> 16);                // add carry
    answer = ~sum;                      // truncate to 16 bits

    return answer;
}
```

In the above function (isValidIpAddress) is used to check if the ip address correct or not, using the function (**inet_pton**) :

This function converts the character string src into a network address structure in the af address family, then copies the network address structure to dst.

The `af` argument must be either `AF_INET` or `AF_INET6`.
`dst` is written in network byte order.

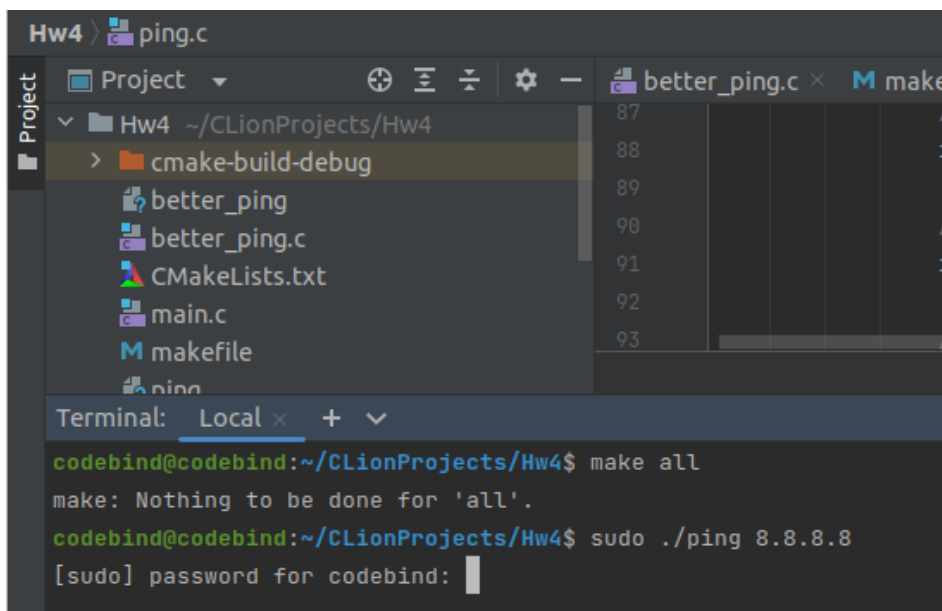
```
bool isValidIpAddress(char *ipAddress)
{
    struct sockaddr_in sa;
    int result = inet_pton(AF_INET, ipAddress, &(sa.sin_addr));
    return result != 0;
}
```

Measure the time of send and get the replay of echo packet, in milliseconds.

```
float milliseconds =
    (end.tv_sec - start.tv_sec) * 1000.0f + (end.tv_usec - start.tv_usec) / 1000.0f;
```

How to run the code:-

- 1.) make all.
- 2.) sudo ./name_file <ip> :-
 sudo ./ping 8.8.8.8
 sudo ./better_ping 8.8.8.8



```
Hw4 / ping.c
Project
  Project
    Hw4 ~/CLionProjects/Hw4
      cmake-build-debug
        better_ping
        better_ping.c
        CMakeLists.txt
        main.c
        makefile
        ping
  better_ping.c x M make
  87
  88
  89
  90
  91
  92
  93
Terminal: Local x + v
codebind@codebind:~/CLionProjects/Hw4$ make all
make: Nothing to be done for 'all'.
codebind@codebind:~/CLionProjects/Hw4$ sudo ./ping 8.8.8.8
[sudo] password for codebind:
```

```
if ( count != 2 )
{
    printf("usage: ./ping <addr> \n");
    exit(-1);
}
```

The above code , is to check if we get true number of arguments in console.

And if we get less or more than what should, then we get an error message and exit the program.

Examples:-

```
shahar@shahar-X442UQR:~/CLionProjects/matala4tikshoret$ sudo ./better_ping
[sudo] password for shahar:
usage: ./better_ping <addr>
```

```
shahar@shahar-X442UQR:~/CLionProjects/matala4tikshoret$ sudo ./ping
usage: ./ping <addr>
```

Or this function

```
bool isValidIpAddress(char *ipAddress)
{
    struct sockaddr_in sa;
    int result = inet_pton(AF_INET, ipAddress, &(sa.sin_addr));
    return result != 0;
}
```

if the ip is not correct then we get an error and close the program.

Examples:-

```
shahar@shahar-X442UQR:~/CLionProjects/matala4tikshoret$ sudo ./ping 8.8
ip address is not Valid
```

כעת, נסביר איך שתי התוכניות עובדות בפועל.

- ping.c

התוכנה שולחת פינג רגיל. התוכנה תקבל שני ארגומנטים בתחילת ההרצה - שם התוכנית (ישמר במערך argv במיקום 0), וכתובת ה-IP אליה אנחנו רוצים לשלוח פינג (יישמר ב-argv במיקום 1). ראשית, אנו פותחים raw socket שדרכו נשלח את בקשת ה-PING. לאחר מכן, בלולאה אינסופית, אנחנו בונים את ה-header של ה-IP ושל ה-ICMP, מצרפים להודעת הפינג ושולחים באמצעות מתודת sendto דרך ה-raw socket. אנחנו בונים את ה-header כל פעם מחדש כי עבור כל שליחה וקבלה, צריך לחשב את ה-checksum. לפני השליחה אנחנו מודדים את הזמן, ובסוף הקבלה מודדים שוב. ככה אנחנו יכולים להגיד כמה זמן לקח לבקשה להישלח ולתגובה להתקבל, ולהדפיס את זה בסוף הקבלה.

בקשת הפינג מכילה כמה פרמטרים:

1. הפרמטר icmp_seq - זהו משתנה שסופר את כל התגובות (pong) שנשלחו תחת קשר socket אחד. כך אנחנו יודעים כמה פעמים התקבלה התגובה לפינג ובעקבות כך כמה פעמים נשלח פינג.
2. הפרמטר ttl - משתנה שמחושב על ידי מערכת ההפעלה ולא על ידינו, שמגדיר את הזמן של החבילה לחיות. ttl יגיד לנו בכמה נתבים החבילה יכולה לעבור עד שיגיע נתב שיזרוק אותה. המשתנה הזה קיים מאחר שלפי החישוב של מערכת ההפעלה, החבילה לא אמורה להגיע למצב שבו בין המקור ליעד היא עוברת ביותר מכמות הנתבים המוגדרת. אם היא הגיעה למצב הזה, כנראה שיש תקלה בשליחה או בניתוב של החבילה, וככל הנראה המידע בה כבר לא רלוונטי. הנתב במקרה זה יעדיף לזרוק אותה ולהגיד לשולח לשלוח חבילה חדשה.
3. הפרמטר time - כמות הזמן במילישניות שלקח לחבילה להישלח ולתגובה להתקבל. בפינג המשודרג הלא הוא better_ping, אם time עובר את ה-10 שניות, הדבר נחשב כ-timeout, כלומר, כנראה שהתגובה גם לא תתקבל בעתיד ואין מה להמשיך לחכות. במצב הזה נסגור את ה-socket ונדפיס הודעה ליוזר שהשרת בלתי ניתן להשגה.

נציין בנוסף שגודל חבילת התגובה היא 64 בייטים. שהם 56 בייטים השייכים לתגובה עצמה ועוד 8 בייטים עבור ה-icmp header.

בפינג הרגיל, אם נשלח פינג למחשב שאינו מחובר - כלומר, נשלח פינג לשרת שלא יחזיר לנו תגובה, התוכנה תחכה בסבלנות עד שתתקבל תגובה עד שאנחנו נקריס אותה. העניין הזה מטופל ב-better ping.

better_ping.c

התוכנה הזו היא תוכנית של פינג משופר. בתוכנית זאת ישלח פינג, ואם לא תתקבל תגובה לאחר 10 שניות, התוכנה תדפיס לנו הודעה "server <ip> cannot be reached". כלומר, השרת הזה כנראה לא מחובר לאינטרנט כי הוא לא שולח הודעת תגובה לפינג.

בתוכנית זאת מעורב כלי חדש בשם watchdog שאנחנו בנינו. ה-watchdog בודק האם לפינג אכן לקח יותר מ-10 שניות להגיע ואם כן מדפיס הודעת שגיאה משלו. מבחינת המימוש, ה-watchdog הוא תהליכון צדדי שאנחנו פותחים מה-better_ping. הוא הולך להתחבר ל-better_ping בחיבור TCP ועל פי האינפורמציה שהוא מקבל מה-better_ping להדפיס האם היה timeout או לא. בתוכנית שלנו, better_ping הוא "צד השרת". כלומר, הוא פותח חיבור ומאזין. ה-watchdog בתור "צד הלקוח" הולך להתחבר ל-better_ping.

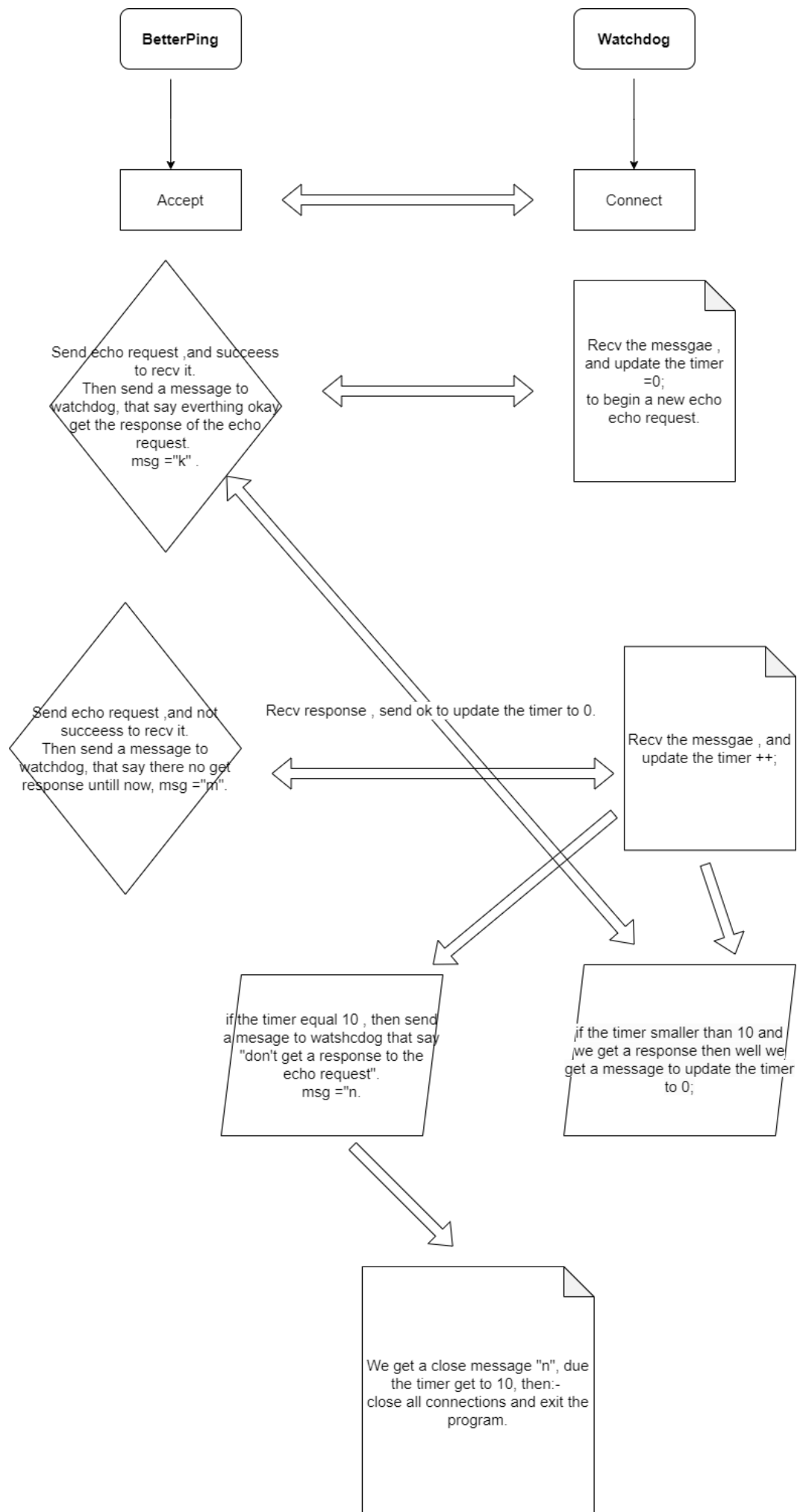
נסביר איך התוכנית עובדת במספר שלבים -

1. better_ping פותח את ה-watchdog בתור תהליכון צדדי על ידי fork ו-exec.
2. better_ping פותח סוקט חדש ומאזין לבקשות חיבור.

3. ה- watchdog יתחבר ל-better_ping בחיבור TCP.
4. better_ping ישלח פינג ל-IP שקיבל בעת ההרצה.
5. better_ping שולח ל-watchdog הודעה שכתוב בה "m". משמעותה עבור ה-watchdog היא "שלחתי הודעה, תתחיל לספור את הזמן".
6. ה-watchdog מקבל את ההודעה, ומתחיל את הטיימר.
7. better_ping מקבל את התגובה (pong). ואז מעדכן את הטיימר לאפס בעת שליחת הודעה "k".
8. בכל שלב שמקבלים תגובה ה better_ping שולח הודעה ל watchdog שכתוב בה "k" ואז מעדכן את הטיימר מחדש לאפס שזה אומר קיבל הודעה חדשה.
9. better_ping שולח ל-watchdog הודעה שכתוב בה "n". משמעותה עבור ה-watchdog היא שהטיימר הגיע ל 10 ואז שולח אותה ל better_ping אחרי קבלה הוא סוגר כל הקישורים וגם התוכנית.
10. כעת, נחזור לשלב 4 ונמשיך כך בצורה אינסופית עד שהיוזר יכבה את התוכנה.

מה קורה כאשר מחשב היעד לא יחזיר לנו תגובה?

נכנס מיד לשלב של הפעלת הטיימר ואז שולח m ל betterping כל עוד עדיין הטיימר לא הגיע ל 10 אם התקבלה תגובה בעת ההרצה של הטיימר אז אנחנו בשלב 8 אחרת הגיע ל 10+ אז בשלב 9 יסגר הכל.



בגזרת ה-WIRESHARK,

הרצת תוכנית PING כאשר השרת מחזיר תשובה:

```
shahar@shahar-X442UQR:~/CLionProjects/matala4tikshoret$ sudo ./ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8): 56 bytes of data
64 bytes from 8.8.8.8: seq=1, ttl=119 time=7.57 ms
64 bytes from 8.8.8.8: seq=2, ttl=119 time=8.34 ms
64 bytes from 8.8.8.8: seq=3, ttl=119 time=6.34 ms
64 bytes from 8.8.8.8: seq=4, ttl=119 time=7.06 ms
64 bytes from 8.8.8.8: seq=5, ttl=119 time=4.51 ms
64 bytes from 8.8.8.8: seq=6, ttl=119 time=22.64 ms
64 bytes from 8.8.8.8: seq=7, ttl=119 time=4.77 ms
64 bytes from 8.8.8.8: seq=8, ttl=119 time=7.30 ms
64 bytes from 8.8.8.8: seq=9, ttl=119 time=4.93 ms
64 bytes from 8.8.8.8: seq=10, ttl=119 time=4.22 ms
64 bytes from 8.8.8.8: seq=11, ttl=119 time=5.72 ms
64 bytes from 8.8.8.8: seq=12, ttl=119 time=82.89 ms
64 bytes from 8.8.8.8: seq=13, ttl=119 time=25.56 ms
64 bytes from 8.8.8.8: seq=14, ttl=119 time=55.15 ms
64 bytes from 8.8.8.8: seq=15, ttl=119 time=8.12 ms
64 bytes from 8.8.8.8: seq=16, ttl=119 time=4.38 ms
64 bytes from 8.8.8.8: seq=17, ttl=119 time=5.83 ms
64 bytes from 8.8.8.8: seq=18, ttl=119 time=6.78 ms
64 bytes from 8.8.8.8: seq=19, ttl=119 time=4.55 ms
64 bytes from 8.8.8.8: seq=20, ttl=119 time=10.58 ms
```

פינג לשרת אשר לא מחזיר תשובה:

```
shahar@shahar-X442UQR:~/CLionProjects/matala4tikshoret$ sudo ./ping 212.179.121.74
PING 212.179.121.74 (212.179.121.74): 56 bytes of data
```

פינג משופר לשרת אשר מחזיר תשובה:

```
shahar@shahar-X442UQR:~/CLionProjects/matala4tikshoret$ sudo ./better_ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8): 56 bytes of data
64 bytes from 8.8.8.8: icmp_seq=1, ttl=119 time=2.22 ms
64 bytes from 8.8.8.8: icmp_seq=2, ttl=119 time=2.19 ms
64 bytes from 8.8.8.8: icmp_seq=3, ttl=119 time=2.38 ms
64 bytes from 8.8.8.8: icmp_seq=4, ttl=119 time=2.42 ms
64 bytes from 8.8.8.8: icmp_seq=5, ttl=119 time=2.24 ms
64 bytes from 8.8.8.8: icmp_seq=6, ttl=119 time=2.40 ms
64 bytes from 8.8.8.8: icmp_seq=7, ttl=119 time=2.60 ms
64 bytes from 8.8.8.8: icmp_seq=8, ttl=119 time=2.19 ms
64 bytes from 8.8.8.8: icmp_seq=9, ttl=119 time=2.23 ms
64 bytes from 8.8.8.8: icmp_seq=10, ttl=119 time=2.34 ms
64 bytes from 8.8.8.8: icmp_seq=11, ttl=119 time=2.27 ms
64 bytes from 8.8.8.8: icmp_seq=12, ttl=119 time=2.23 ms
64 bytes from 8.8.8.8: icmp_seq=13, ttl=119 time=2.26 ms
64 bytes from 8.8.8.8: icmp_seq=14, ttl=119 time=2.28 ms
64 bytes from 8.8.8.8: icmp_seq=15, ttl=119 time=2.20 ms
64 bytes from 8.8.8.8: icmp_seq=16, ttl=119 time=2.40 ms
64 bytes from 8.8.8.8: icmp_seq=17, ttl=119 time=2.23 ms
64 bytes from 8.8.8.8: icmp_seq=18, ttl=119 time=2.29 ms
64 bytes from 8.8.8.8: icmp_seq=19, ttl=119 time=2.59 ms
64 bytes from 8.8.8.8: icmp_seq=20, ttl=119 time=2.38 ms
```

פינג משופר לשרת אשר לא מחזיר תשובה:

```
shahar@shahar-X442UQR:~/CLionProjects/matala4tikshoret$ sudo ./better_ping 212.179.121.74
PING 212.179.121.74 (212.179.121.74): 56 bytes of data
server <212.179.121.74> cannot be reached

--- 212.179.121.74 ping statistics ---
10 packets transmitted, 0 received, 100% packet loss, time 100000.000000ms
```


ובגזרת ה-WIRESHARK,

מצורפות למטלה 4 הקלטות:

פינג רגיל לגוגל - ping_to_connected_server:

ping_to_connected_server.pcapng						
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help						
icmp						
No.	Source	Destination	Protocol	Length	Info	
1	10.9.7.253	8.8.8.8	ICMP	62	Echo (ping) request	id=0x1200, seq=0/0, ttl=64 (reply in 2)
2	8.8.8.8	10.9.7.253	ICMP	62	Echo (ping) reply	id=0x1200, seq=0/0, ttl=119 (request in 1)
3	10.9.7.253	8.8.8.8	ICMP	62	Echo (ping) request	id=0x1200, seq=256/1, ttl=64 (reply in 4)
4	8.8.8.8	10.9.7.253	ICMP	62	Echo (ping) reply	id=0x1200, seq=256/1, ttl=119 (request in 3)
5	10.9.7.253	8.8.8.8	ICMP	62	Echo (ping) request	id=0x1200, seq=512/2, ttl=64 (reply in 6)
6	8.8.8.8	10.9.7.253	ICMP	62	Echo (ping) reply	id=0x1200, seq=512/2, ttl=119 (request in 5)
7	10.9.7.253	8.8.8.8	ICMP	62	Echo (ping) request	id=0x1200, seq=768/3, ttl=64 (reply in 8)
8	8.8.8.8	10.9.7.253	ICMP	62	Echo (ping) reply	id=0x1200, seq=768/3, ttl=119 (request in 7)
9	10.9.7.253	8.8.8.8	ICMP	62	Echo (ping) request	id=0x1200, seq=1024/4, ttl=64 (reply in 10)
10	8.8.8.8	10.9.7.253	ICMP	62	Echo (ping) reply	id=0x1200, seq=1024/4, ttl=119 (request in 9)
11	10.9.7.253	8.8.8.8	ICMP	62	Echo (ping) request	id=0x1200, seq=1280/5, ttl=64 (reply in 12)
12	8.8.8.8	10.9.7.253	ICMP	62	Echo (ping) reply	id=0x1200, seq=1280/5, ttl=119 (request in 11)
13	10.9.7.253	8.8.8.8	ICMP	62	Echo (ping) request	id=0x1200, seq=1536/6, ttl=64 (reply in 14)
14	8.8.8.8	10.9.7.253	ICMP	62	Echo (ping) reply	id=0x1200, seq=1536/6, ttl=119 (request in 13)
15	10.9.7.253	8.8.8.8	ICMP	62	Echo (ping) request	id=0x1200, seq=1792/7, ttl=64 (reply in 16)
16	8.8.8.8	10.9.7.253	ICMP	62	Echo (ping) reply	id=0x1200, seq=1792/7, ttl=119 (request in 15)
17	10.9.7.253	8.8.8.8	ICMP	62	Echo (ping) request	id=0x1200, seq=2048/8, ttl=64 (reply in 18)
18	8.8.8.8	10.9.7.253	ICMP	62	Echo (ping) reply	id=0x1200, seq=2048/8, ttl=119 (request in 17)
19	10.9.7.253	8.8.8.8	ICMP	62	Echo (ping) request	id=0x1200, seq=2304/9, ttl=64 (reply in 20)
20	8.8.8.8	10.9.7.253	ICMP	62	Echo (ping) reply	id=0x1200, seq=2304/9, ttl=119 (request in 19)
21	10.9.7.253	8.8.8.8	ICMP	62	Echo (ping) request	id=0x1200, seq=2560/10, ttl=64 (reply in 22)
22	8.8.8.8	10.9.7.253	ICMP	62	Echo (ping) reply	id=0x1200, seq=2560/10, ttl=119 (request in 21)
23	10.9.7.253	8.8.8.8	ICMP	62	Echo (ping) request	id=0x1200, seq=2816/11, ttl=64 (reply in 24)
24	8.8.8.8	10.9.7.253	ICMP	62	Echo (ping) reply	id=0x1200, seq=2816/11, ttl=119 (request in 23)
25	10.9.7.253	8.8.8.8	ICMP	62	Echo (ping) request	id=0x1200, seq=3072/12, ttl=64 (reply in 26)
26	8.8.8.8	10.9.7.253	ICMP	62	Echo (ping) reply	id=0x1200, seq=3072/12, ttl=119 (request in 25)
27	10.9.7.253	8.8.8.8	ICMP	62	Echo (ping) request	id=0x1200, seq=3328/13, ttl=64 (reply in 28)
28	8.8.8.8	10.9.7.253	ICMP	62	Echo (ping) reply	id=0x1200, seq=3328/13, ttl=119 (request in 27)
29	10.9.7.253	8.8.8.8	ICMP	62	Echo (ping) request	id=0x1200, seq=3584/14, ttl=64 (reply in 30)
30	8.8.8.8	10.9.7.253	ICMP	62	Echo (ping) reply	id=0x1200, seq=3584/14, ttl=119 (request in 29)
31	10.9.7.253	8.8.8.8	ICMP	62	Echo (ping) request	id=0x1200, seq=3840/15, ttl=64 (reply in 32)
32	8.8.8.8	10.9.7.253	ICMP	62	Echo (ping) reply	id=0x1200, seq=3840/15, ttl=119 (request in 31)
33	10.9.7.253	8.8.8.8	ICMP	62	Echo (ping) request	id=0x1200, seq=4096/16, ttl=64 (reply in 34)
34	8.8.8.8	10.9.7.253	ICMP	62	Echo (ping) reply	id=0x1200, seq=4096/16, ttl=119 (request in 33)
35	10.9.7.253	8.8.8.8	ICMP	62	Echo (ping) request	id=0x1200, seq=4352/17, ttl=64 (reply in 36)
36	8.8.8.8	10.9.7.253	ICMP	62	Echo (ping) reply	id=0x1200, seq=4352/17, ttl=119 (request in 35)
37	10.9.7.253	8.8.8.8	ICMP	62	Echo (ping) request	id=0x1200, seq=4608/18, ttl=64 (reply in 38)
38	8.8.8.8	10.9.7.253	ICMP	62	Echo (ping) reply	id=0x1200, seq=4608/18, ttl=119 (request in 37)
39	10.9.7.253	8.8.8.8	ICMP	62	Echo (ping) request	id=0x1200, seq=4864/19, ttl=64 (reply in 40)
40	8.8.8.8	10.9.7.253	ICMP	62	Echo (ping) reply	id=0x1200, seq=4864/19, ttl=119 (request in 39)
Internet Control Message Protocol: Protocol						
				Packets: 40 - Displayed: 40 (100.0%)		Profile: Default

נוכל לראות בהקלטה זו, שיש 20 חבילות של בקשה, ו-20 חבילות של תגובה מהשרת. המחשב שלי שולח הודעה ומחכה עד שתגיע תגובה, וממשיך ככה בלולאה אינסופית עד שמקריסים את התוכנה. נוכל לראות עוד דבר נחמד - הגדרנו הודעה שאנחנו שולחים בתוך הפינג והיא "this is the ping" ואם נסתכל בתוך פקטה ב-WIRESHARK, נוכל לראות אותה כתובה ב-WIRESHARK.

Wireshark · Packet 11 · better_ping_to_connected_server.pcapng

- ▶ Frame 11: 62 bytes on wire (496 bits), 62 bytes captured (496 bits) on interface any, id 0
- ▶ Linux cooked capture v1
- ▶ Internet Protocol Version 4, Src: 8.8.8.8, Dst: 10.9.7.253
- ▶ Internet Control Message Protocol

0000	00 00 00 01 00 06 40 b5 c1 f5 ae 3c 4a 9e 08 00@. ...<J...
0010	45 60 00 2e 00 00 00 00 77 01 21 5a 08 08 08 08	E`.....w!Z....
0020	0a 09 07 fd 00 00 b6 36 12 00 00 00 54 68 69 736This
0030	20 69 73 20 74 68 65 20 70 69 6e 67 2e 0a	is the ping..

Help

Close

פינג רגיל לשרת שאינו מחובר לאינטרנט - ping_to_disconnected_server:

ping_to_disconnected_server.pcapng				
No.	Source	Destination	Protocol	Length Info
1	10.9.7.253	212.179.121.74	ICMP	62 Echo (ping) request id=0x1200, seq=0/0, ttl=64 (no response found!)

כאן, אנחנו שולחים בקשת פינג לשרת שלא הולך להחזיר לנו תשובה. תוכנת הפינג ברמת העיקרון יכולה להמשיך להמתין ככה לנצח לתגובה מהשרת, עד שנקריס אותה בעצמנו. לכן, הקלטת ה-WIRESHARK במקרה זה מכילה רק חבילה אחת, חבילת הבקשה הראשונה.

פינג משופר לגוגל - better_ping_to_connected_server:

better_ping_to_connected_server.pcapng				
icmp (ip.src == 127.0.0.1 && ip.dst == 127.0.0.1)				
No.	Source	Destination	Protocol	Length Info
1	127.0.0.1	127.0.0.1	TCP	76 40170 - 3000 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=2666362522 TSecr=0 WS=128
2	127.0.0.1	127.0.0.1	TCP	76 3000 - 40170 [SYN, ACK] Seq=0 Ack=1 Win=65463 Len=0 MSS=65495 SACK_PERM=1 TSval=2666362522 TSecr=2666363522
3	127.0.0.1	127.0.0.1	TCP	68 40170 - 3000 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=2666362522 TSecr=2666362522
4	10.9.7.253	8.8.8.8	ICMP	62 Echo (ping) request id=0x1200, seq=0/0, ttl=64 (reply in 5)
5	8.8.8.8	10.9.7.253	ICMP	62 Echo (ping) reply id=0x1200, seq=0/0, ttl=119 (request in 4)
6	127.0.0.1	127.0.0.1	TCP	70 40170 - 3000 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=2 TSval=2666363522 TSecr=2666362522
7	127.0.0.1	127.0.0.1	TCP	68 3000 - 40170 [ACK] Seq=1 Ack=3 Win=65536 Len=0 TSval=2666363522 TSecr=2666363522
8	10.9.7.253	8.8.8.8	ICMP	62 Echo (ping) request id=0x1200, seq=0/0, ttl=64 (reply in 11)
9	127.0.0.1	127.0.0.1	TCP	69 3000 - 40170 [PSH, ACK] Seq=1 Ack=3 Win=65536 Len=1 TSval=2666363524 TSecr=2666363522
10	127.0.0.1	127.0.0.1	TCP	68 40170 - 3000 [ACK] Seq=3 Ack=2 Win=65536 Len=0 TSval=2666363524 TSecr=2666363524
11	8.8.8.8	10.9.7.253	ICMP	62 Echo (ping) reply id=0x1200, seq=0/0, ttl=119 (request in 8)
12	127.0.0.1	127.0.0.1	TCP	70 40170 - 3000 [PSH, ACK] Seq=3 Ack=2 Win=65536 Len=2 TSval=2666364522 TSecr=2666363524
13	10.9.7.253	8.8.8.8	ICMP	62 Echo (ping) request id=0x1200, seq=256/1, ttl=64 (reply in 16)
14	127.0.0.1	127.0.0.1	TCP	69 3000 - 40170 [PSH, ACK] Seq=2 Ack=5 Win=65536 Len=1 TSval=2666364526 TSecr=2666364522
15	127.0.0.1	127.0.0.1	TCP	68 40170 - 3000 [ACK] Seq=5 Ack=3 Win=65536 Len=0 TSval=2666364526 TSecr=2666364526
16	8.8.8.8	10.9.7.253	ICMP	62 Echo (ping) reply id=0x1200, seq=256/1, ttl=119 (request in 13)
17	127.0.0.1	127.0.0.1	TCP	70 40170 - 3000 [PSH, ACK] Seq=5 Ack=3 Win=65536 Len=2 TSval=2666365522 TSecr=2666364526
18	10.9.7.253	8.8.8.8	ICMP	62 Echo (ping) request id=0x1200, seq=512/2, ttl=64 (reply in 21)
19	127.0.0.1	127.0.0.1	TCP	69 3000 - 40170 [PSH, ACK] Seq=3 Ack=7 Win=65536 Len=1 TSval=2666365529 TSecr=2666365522
20	127.0.0.1	127.0.0.1	TCP	68 40170 - 3000 [ACK] Seq=7 Ack=4 Win=65536 Len=0 TSval=2666365529 TSecr=2666365529
21	8.8.8.8	10.9.7.253	ICMP	62 Echo (ping) reply id=0x1200, seq=512/2, ttl=119 (request in 18)
22	127.0.0.1	127.0.0.1	TCP	70 40170 - 3000 [PSH, ACK] Seq=7 Ack=4 Win=65536 Len=2 TSval=2666366523 TSecr=2666365529
23	10.9.7.253	8.8.8.8	ICMP	62 Echo (ping) request id=0x1200, seq=768/3, ttl=64 (reply in 26)
24	127.0.0.1	127.0.0.1	TCP	69 3000 - 40170 [PSH, ACK] Seq=4 Ack=9 Win=65536 Len=1 TSval=2666366532 TSecr=2666366523
25	127.0.0.1	127.0.0.1	TCP	68 40170 - 3000 [ACK] Seq=9 Ack=5 Win=65536 Len=0 TSval=2666366532 TSecr=2666366532
26	8.8.8.8	10.9.7.253	ICMP	62 Echo (ping) reply id=0x1200, seq=768/3, ttl=119 (request in 23)
27	127.0.0.1	127.0.0.1	TCP	70 40170 - 3000 [PSH, ACK] Seq=9 Ack=5 Win=65536 Len=2 TSval=2666367523 TSecr=2666366532
28	10.9.7.253	8.8.8.8	ICMP	62 Echo (ping) request id=0x1200, seq=1024/4, ttl=64 (reply in 31)
29	127.0.0.1	127.0.0.1	TCP	69 3000 - 40170 [PSH, ACK] Seq=5 Ack=11 Win=65536 Len=1 TSval=2666367534 TSecr=2666367523
30	127.0.0.1	127.0.0.1	TCP	68 40170 - 3000 [ACK] Seq=11 Ack=6 Win=65536 Len=0 TSval=2666367534 TSecr=2666367534
31	8.8.8.8	10.9.7.253	ICMP	62 Echo (ping) reply id=0x1200, seq=1024/4, ttl=119 (request in 29)
32	127.0.0.1	127.0.0.1	TCP	70 40170 - 3000 [PSH, ACK] Seq=11 Ack=6 Win=65536 Len=2 TSval=2666368523 TSecr=2666367534
33	10.9.7.253	8.8.8.8	ICMP	62 Echo (ping) request id=0x1200, seq=1280/5, ttl=64 (reply in 36)
34	127.0.0.1	127.0.0.1	TCP	69 3000 - 40170 [PSH, ACK] Seq=6 Ack=13 Win=65536 Len=1 TSval=2666368536 TSecr=2666368523
35	127.0.0.1	127.0.0.1	TCP	68 40170 - 3000 [ACK] Seq=13 Ack=7 Win=65536 Len=0 TSval=2666368536 TSecr=2666368536
36	8.8.8.8	10.9.7.253	ICMP	62 Echo (ping) reply id=0x1200, seq=1280/5, ttl=119 (request in 33)
37	127.0.0.1	127.0.0.1	TCP	70 40170 - 3000 [PSH, ACK] Seq=13 Ack=7 Win=65536 Len=2 TSval=2666369523 TSecr=2666368536
38	10.9.7.253	8.8.8.8	ICMP	62 Echo (ping) request id=0x1200, seq=1536/6, ttl=64 (reply in 41)
39	127.0.0.1	127.0.0.1	TCP	69 3000 - 40170 [PSH, ACK] Seq=7 Ack=15 Win=65536 Len=1 TSval=2666369539 TSecr=2666369523
40	127.0.0.1	127.0.0.1	TCP	68 40170 - 3000 [ACK] Seq=15 Ack=8 Win=65536 Len=0 TSval=2666369539 TSecr=2666369539
41	8.8.8.8	10.9.7.253	ICMP	62 Echo (ping) reply id=0x1200, seq=1536/6, ttl=119 (request in 38)
42	127.0.0.1	127.0.0.1	TCP	70 40170 - 3000 [PSH, ACK] Seq=15 Ack=8 Win=65536 Len=2 TSval=2666370523 TSecr=2666369539
43	10.9.7.253	8.8.8.8	ICMP	62 Echo (ping) request id=0x1200, seq=1792/7, ttl=64 (reply in 46)
44	127.0.0.1	127.0.0.1	TCP	69 3000 - 40170 [PSH, ACK] Seq=8 Ack=17 Win=65536 Len=1 TSval=2666370542 TSecr=2666370523
45	127.0.0.1	127.0.0.1	TCP	68 40170 - 3000 [ACK] Seq=17 Ack=8 Win=65536 Len=0 TSval=2666370542 TSecr=2666370542

כאן כבר נהיה קצת יותר מעניין. כאן יש צילום של חלק מההקלטה, כדי להראות את העיקרון, למרות ששליחה של 20 בקשות פינג מצריכה מאיתנו 109 פקטות.

נוכל לראות שבהתחלת התקשורת, יש את שתי ההודעות המסומנות באפור שמעידות על פתיחת החיבור בין ה-better-ping ל-watchdog. לאחר מכן, לאחר כל חבילת שליחה או חבילת קבלה יש גם חבילות של PSH ל-watchdog. אלה בדיוק ההודעות שאנחנו שולחים ל-watchdog שאומרות לו להמשיך/להפסיק/לאתחל את הטיימר.

פינג משופר לשרת שאינו מחובר לאינטרנט - better_ping_to_disconnected_server

better_ping_to_disconnected_server.pcapng					
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help					
icmp (ip.src == 127.0.0.1 && ip.dst == 127.0.0.1)					
No.	Source	Destination	Protocol	Length	Info
1	127.0.0.1	127.0.0.1	TCP	76	47898 → 3000 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=2666637889 TSecr=0 WS=128
2	127.0.0.1	127.0.0.1	TCP	76	3000 → 47898 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=2666637889 TSecr=2666637889
3	127.0.0.1	127.0.0.1	TCP	68	47898 → 3000 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=2666637889 TSecr=2666637889
4	10.0.7.253	212.179.121.74	ICMP	62	Echo (ping) request id=0x1200, seq=0/0, ttl=64 (no response found)
5	127.0.0.1	127.0.0.1	TCP	70	47898 → 3000 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=2 TSval=2666638089 TSecr=2666637889
6	127.0.0.1	127.0.0.1	TCP	68	3000 → 47898 [ACK] Seq=1 Ack=3 Win=65536 Len=0 TSval=2666638089 TSecr=2666638089
7	10.0.7.253	212.179.121.74	ICMP	62	Echo (ping) request id=0x1200, seq=0/0, ttl=64 (no response found)
8	127.0.0.1	127.0.0.1	TCP	70	47898 → 3000 [PSH, ACK] Seq=3 Ack=1 Win=65536 Len=2 TSval=2666639089 TSecr=2666638089
9	127.0.0.1	127.0.0.1	TCP	68	3000 → 47898 [ACK] Seq=1 Ack=5 Win=65536 Len=0 TSval=2666639089 TSecr=2666639089
10	10.0.7.253	212.179.121.74	ICMP	62	Echo (ping) request id=0x1200, seq=0/0, ttl=64 (no response found)
11	127.0.0.1	127.0.0.1	TCP	70	47898 → 3000 [PSH, ACK] Seq=5 Ack=1 Win=65536 Len=2 TSval=2666640090 TSecr=2666639089
12	127.0.0.1	127.0.0.1	TCP	68	3000 → 47898 [ACK] Seq=1 Ack=7 Win=65536 Len=0 TSval=2666640090 TSecr=2666640090
13	127.0.0.1	127.0.0.1	TCP	70	47898 → 3000 [PSH, ACK] Seq=7 Ack=1 Win=65536 Len=2 TSval=2666641090 TSecr=2666640090
14	127.0.0.1	127.0.0.1	TCP	68	3000 → 47898 [ACK] Seq=1 Ack=9 Win=65536 Len=0 TSval=2666641090 TSecr=2666641090
15	10.0.7.253	212.179.121.74	ICMP	62	Echo (ping) request id=0x1200, seq=0/0, ttl=64 (no response found)
16	127.0.0.1	127.0.0.1	TCP	70	47898 → 3000 [PSH, ACK] Seq=9 Ack=1 Win=65536 Len=2 TSval=2666642090 TSecr=2666641090
17	127.0.0.1	127.0.0.1	TCP	68	3000 → 47898 [ACK] Seq=1 Ack=11 Win=65536 Len=0 TSval=2666642090 TSecr=2666642090
18	10.0.7.253	212.179.121.74	ICMP	62	Echo (ping) request id=0x1200, seq=0/0, ttl=64 (no response found)
19	127.0.0.1	127.0.0.1	TCP	70	47898 → 3000 [PSH, ACK] Seq=11 Ack=1 Win=65536 Len=2 TSval=2666643090 TSecr=2666642090
20	127.0.0.1	127.0.0.1	TCP	68	3000 → 47898 [ACK] Seq=1 Ack=13 Win=65536 Len=0 TSval=2666643090 TSecr=2666643090
21	10.0.7.253	212.179.121.74	ICMP	62	Echo (ping) request id=0x1200, seq=0/0, ttl=64 (no response found)
22	127.0.0.1	127.0.0.1	TCP	70	47898 → 3000 [PSH, ACK] Seq=13 Ack=1 Win=65536 Len=2 TSval=2666644090 TSecr=2666643090
23	127.0.0.1	127.0.0.1	TCP	68	3000 → 47898 [ACK] Seq=1 Ack=15 Win=65536 Len=0 TSval=2666644090 TSecr=2666644090
24	10.0.7.253	212.179.121.74	ICMP	62	Echo (ping) request id=0x1200, seq=0/0, ttl=64 (no response found)
25	127.0.0.1	127.0.0.1	TCP	70	47898 → 3000 [PSH, ACK] Seq=15 Ack=1 Win=65536 Len=2 TSval=2666645091 TSecr=2666644090
26	127.0.0.1	127.0.0.1	TCP	68	3000 → 47898 [ACK] Seq=1 Ack=17 Win=65536 Len=0 TSval=2666645091 TSecr=2666645091
27	10.0.7.253	212.179.121.74	ICMP	62	Echo (ping) request id=0x1200, seq=0/0, ttl=64 (no response found)
28	127.0.0.1	127.0.0.1	TCP	70	47898 → 3000 [PSH, ACK] Seq=17 Ack=1 Win=65536 Len=2 TSval=2666646091 TSecr=2666645091
29	127.0.0.1	127.0.0.1	TCP	68	3000 → 47898 [ACK] Seq=1 Ack=19 Win=65536 Len=0 TSval=2666646091 TSecr=2666646091
30	10.0.7.253	212.179.121.74	ICMP	62	Echo (ping) request id=0x1200, seq=0/0, ttl=64 (no response found)
31	127.0.0.1	127.0.0.1	TCP	70	47898 → 3000 [PSH, ACK] Seq=19 Ack=1 Win=65536 Len=2 TSval=2666647091 TSecr=2666646091
32	127.0.0.1	127.0.0.1	TCP	68	3000 → 47898 [ACK] Seq=1 Ack=21 Win=65536 Len=0 TSval=2666647091 TSecr=2666647091
33	127.0.0.1	127.0.0.1	TCP	68	47898 → 3000 [PSH, ACK] Seq=21 Ack=1 Win=65536 Len=0 TSval=2666647891 TSecr=2666647091
34	10.0.7.253	212.179.121.74	ICMP	62	Echo (ping) request id=0x1200, seq=0/0, ttl=64 (no response found)
35	127.0.0.1	127.0.0.1	TCP	68	3000 → 47898 [RST, ACK] Seq=1 Ack=22 Win=65536 Len=0 TSval=2666647110 TSecr=2666647891

נוכל לראות גם כאן שבתחילת התקשורת יש 2 הודעות המעידות על פתיחת החיבור בין ה-better_ping- ל-watchdog. בהקלטה, נוכל לראות שיש 10 חבילות בקשה שנשלחות לשרת, מאחר שאנחנו שולחים פינג כל שנייה. בניגוד לפינג הרגיל, כאן הגדרנו את פונקציית RECV להיות NON-BLOCKING - כלומר, אם ה-RECV לא מקבל מידע, הוא לא ממשיך לחכות למידע, והתוכנית תמשיך הלאה. לכן, אנחנו מצליחים לשלוח 10 בקשות פינג ולא שולחים רק אחת ומככים לתשובה. לבסוף, לאחר שהטיימר הגיע ל-10 שניות מכבים את התוכנה, וסוגרים את הסוקט. נשים לב שלפני ההודעה האדומה ישנה הודעה אפורה שמעידה על סגירת החיבור מהצד של watchdog. ואנחנו אכן סגרנו את החיבור מהצד של watchdog ללא ידיעתו של הצד השני, better_ping. לכן מופיעה ההודעה האדומה.

נשים לב, שהמחשב קולט שלא מתקבלת תגובה כי רואים את זה ב-wireshark בהודעה "no response found".