Alon Meshulam - 207964487
Israel Gitler - 208580076

Readme - Ping and Watchdog

Software Execution

The software is designed to run on the Ubuntu 22.04LTS operating system with Python 3.10. To run the software, follow these steps:

1. Open the project on Ubuntu 22.04LTS.
2. Create a local terminal.
3. Run: sudo python3 <ping/better_ping>.py <Ip address>

System Overview

The system is a network monitoring and diagnostics tool that performs ICMP ping to a specified IP address and provides real-time information about the network connectivity. It includes an improved version of the ping functionality with a watchdog feature to monitor and detect network connectivity issues.

System Functionality

1. Perform ICMP ping to a specified IP address.
2. Calculate the checksum of ICMP packets.
3. Send ICMP Echo Request packets and wait for replies.
4. Handle ICMP replies and process the received data.
5. Print ping results to the console.
6. Watchdog functionality to monitor the ping process and detect timeouts.
7. Notify the watchdog when ping replies are received, or errors occur.
8. Monitor network connectivity and alert if the destination IP is unreachable.

Components and Subsystems

The system consists of the following components and subsystems:

1. **ping.py:** Implements the ICMP ping functionality.
2. **better_ping.py:** Includes the improved version of the ping functionality with the watchdog feature.

3. **watchdog.py:** Implements the watchdog functionality to monitor the ping process.
4. **ICMP (Internet Control Message Protocol):** Used for sending and receiving ICMP packets.
5. **TCP (Transmission Control Protocol):** Used for communication between better_ping.py and watchdog.py through a TCP socket.

Threat Sources

1. Network connectivity issues: Unstable or unreliable network connections can cause ping timeouts or incorrect results.
2. IP spoofing: Attackers may attempt to forge the source IP address in ICMP packets, leading to false or misleading ping results.

Alon Meshulam - 207964487
Israel Gitler - 208580076

Describing Ping and Watchdog (in general)

Ping: Ping is a network utility that measures the round-trip time (RTT) for data packets sent from a source device to a destination device and back. It is commonly used to check the reachability and response time of a network host or IP address.

Key Features of Ping:

- ICMP Packet Creation: Ping generates ICMP Echo Request packets.
- Packet Transmission: The ICMP Echo Request packets are sent from the source device to the destination IP address using a raw socket.
- Packet Reception and Processing: Upon receiving ICMP Echo Reply packets, the system extracts relevant information such as packet size, source IP, sequence number, and response time. It verifies the packet's type and sequence number for validity.
- Result Presentation: Ping displays the received ping reply information, including packet size, source IP, sequence number, TTL (Time to Live), and response time in milliseconds. It also handles scenarios where the destination is unreachable or a request times out.
- Continuous Pinging: Ping facilitates continuous monitoring by sending ICMP Echo Request packets with incrementing sequence numbers and waiting for corresponding replies. It provides a mechanism to stop the pinging process upon user interruption.

Watchdog:

Watchdog is a timer to detect and recover your computer dis-functions or hardware fails. It’s a chip whose sole purpose is to receive a signal every millisecond from the CPU. It will reboot the system if it hasn’t received any signal for 10 milliseconds (mostly when hardware fails).

Alon Meshulam - 207964487
Israel Gitler - 208580076

References

1. Ping (networking utility): Wikipedia
2. ICMP (Internet Control Message Protocol): RFC 792
3. Python socket programming: Python documentation
4. TCP/IP Sockets: GeeksforGeeks
5. Internet Control Message Protocol (ICMP): Cisco documentation

Alon Meshulam - 207964487
Israel Gitler - 208580076

Code Overview

1. **ping.py**:

- Purpose: This file contains the implementation of ICMP ping functionality to send echo requests to a specified IP address and receive echo replies.
- Macro Process:
    1. The program resolves the IP address of the destination using **socket.gethostbyname**.
    2. It creates a raw socket with **socket.socket** and sets the socket options.
    3. The program enters a loop to send echo requests and receive echo replies continuously until interrupted by a keyboard interrupt.
    4. Inside the loop, it constructs an ICMP echo request packet using **create_packet**.
    5. The packet is sent to the destination using **sendto**.
    6. It waits for the corresponding echo reply using **receive_ping** and processes the received reply.
- Key Functions:
    - **calculate_checksum(packet)**: Calculates the checksum of the given packet using an algorithm.
    - **create_packet(seq_num)**: Creates an ICMP echo request packet.
    - **receive_ping(sock, seq_num)**: Receives and processes the echo reply from the socket.
    - **send_ping(sock, dest_addr, seq_num)**: Sends an ICMP echo request packet to the destination address and waits for a reply.

- **ping(ip)**: Performs ICMP ping to the specified IP address by continuously sending echo requests and receiving replies.

2. better_ping.py:

- Purpose: This file extends the functionality of **ping.py** by introducing a watchdog mechanism. It sends updates to the watchdog program to indicate successful replies or timeouts.
- Macro Process:
    1. Same as ping.py process.
    2. It starts a separate thread for the watchdog functionality using **threading.Thread** and **open_watchdog_socket**.
    3. The program enters a loop to send echo requests and receive echo replies continuously until interrupted by a keyboard interrupt or a stop signal from the watchdog.
    4. Inside the loop, it constructs an ICMP echo request packet using **create_packet**.
    5. The packet is sent to the destination using **sendto**.
    6. It waits for the corresponding echo reply using **receive_ping** and processes the received reply.
    7. If a reply is received, it sends an update to the watchdog indicating success.
- Key Functions: The key functions in **better_ping.py** are the same as those in **ping.py**, with the addition of the following function:
    - **open_watchdog_socket(ip)**: Opens a TCP socket for the watchdog functionality.

Alon Meshulam - 207964487
Israel Gitler - 208580076

3. watchdog.py:

- Purpose: This file implements the watchdog functionality that monitors the updates from the **better_ping** program to detect timeouts and failures.
- Macro Process:
    1. The program creates a TCP socket with **socket.socket**, binds it to a specific port (3000), and starts listening for connections.
    2. It accepts an incoming connection from the **better_ping** program using **sock.accept**.
    3. It starts a timer using **time.time** to track the elapsed time.
    4. Inside a loop, it receives updates from the **better_ping** program indicating successful replies or timeouts.
    5. If a successful reply update is received, it resets the timer.
    6. If "did not get answer" update is received, it compares the elapsed time with a predefined timeout (10) to determine if a failure occurred.
    7. If a failure is detected, it close the program and the sockets.
- Key Functions:
    - **open_watchdog_socket()**: Starts the watchdog functionality by creating a TCP socket, accepting connections, and monitoring updates from the **better_ping** program

Alon Meshulam - 207964487
Israel Gitler - 208580076

מכיוון שהחל מעתה הכתיבה היא פחות פורמלית ויותר "הסברתית", ברשותכם נעבור לעברית.

חלק א

בחלק זה השתמשנו בממשק enp0s3 המאפשר לזהות תעבורת ICMP באופן חיצוני. כמו כן, בחלק זה סיננו את ההסנפות לפי הפילטר ICMP.

בפעם הראשונה הרצנו את ping.py עם כתובת ה-IP 1.2.3.4 וקיבלנו את הפלט הבא:

```
alon@alonm:~/PycharmProjects/net4$ sudo python3 ping.py 1.2.3.4
PING 1.2.3.4
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
^CPing stopped by Ctrl-c
```

כתובת IP זו אינה קיימת ועל כן לא ציפינו לקבל תשובה (Pong) כאשר הגדרנו בקוד Timeout בן שנייה לקבלת תשובה. לאחר זמן קצר עצרנו את הריצה באמצעות Ctrl+c.

להלן תיעוד ההסנפה המלא:

icmp

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	10.0.2.15	1.2.3.4	ICMP	46	Echo (ping) request id=0x2a33, seq=256/1, ttl=64 (no response …
2	2.007973408	10.0.2.15	1.2.3.4	ICMP	46	Echo (ping) request id=0x2a33, seq=512/2, ttl=64 (no response …
11	4.011211552	10.0.2.15	1.2.3.4	ICMP	46	Echo (ping) request id=0x2a33, seq=768/3, ttl=64 (no response …
12	6.016236225	10.0.2.15	1.2.3.4	ICMP	46	Echo (ping) request id=0x2a33, seq=1024/4, ttl=64 (no response…
13	8.025733353	10.0.2.15	1.2.3.4	ICMP	46	Echo (ping) request id=0x2a33, seq=1280/5, ttl=64 (no response…
14	10.035010569	10.0.2.15	1.2.3.4	ICMP	46	Echo (ping) request id=0x2a33, seq=1536/6, ttl=64 (no response…
15	12.046168544	10.0.2.15	1.2.3.4	ICMP	46	Echo (ping) request id=0x2a33, seq=1792/7, ttl=64 (no response…

ניתן לראות בהסנפה שבע בקשות שלא נענות.

--

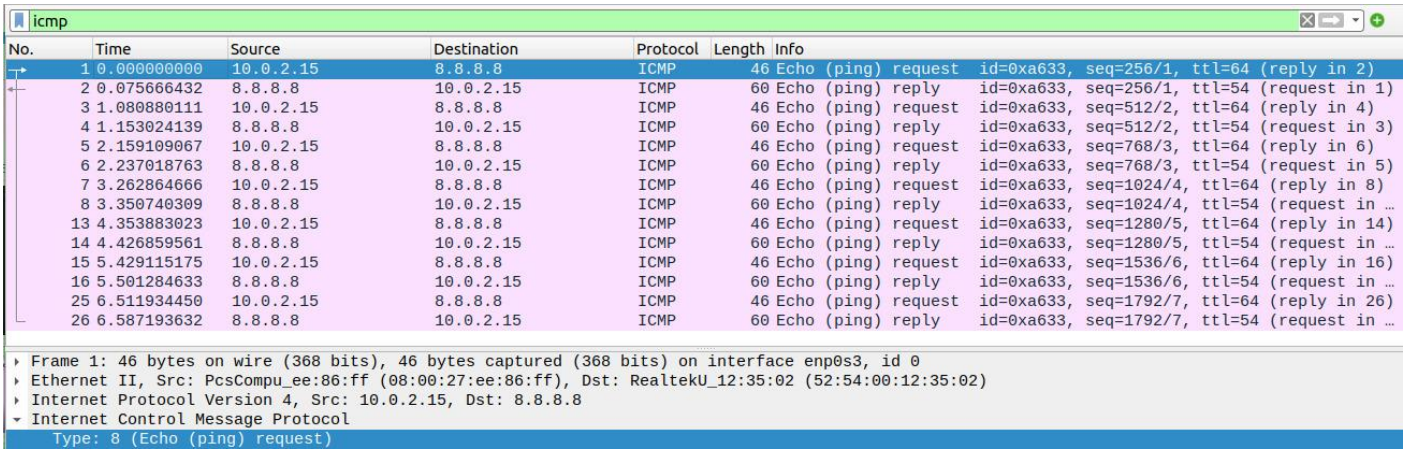
בפעם השנייה הרצנו את ping.py עם כתובת ה-IP 8.8.8.8 (Google) וקיבלנו את הפלט הבא:

Alon Meshulam - 207964487
Israel Gitler - 208580076

```
alon@alonm:~/PycharmProjects/net4$ sudo python3 ping.py 8.8.8.8
PING 8.8.8.8
32 bytes from 8.8.8.8 icmp_seq=1 ttl=54 time=76.441 ms
32 bytes from 8.8.8.8 icmp_seq=2 ttl=54 time=71.844 ms
32 bytes from 8.8.8.8 icmp_seq=3 ttl=54 time=78.087 ms
32 bytes from 8.8.8.8 icmp_seq=4 ttl=54 time=88.202 ms
32 bytes from 8.8.8.8 icmp_seq=5 ttl=54 time=73.144 ms
32 bytes from 8.8.8.8 icmp_seq=6 ttl=54 time=72.430 ms
32 bytes from 8.8.8.8 icmp_seq=7 ttl=54 time=75.352 ms
^CPing stopped by Ctrl-c
```

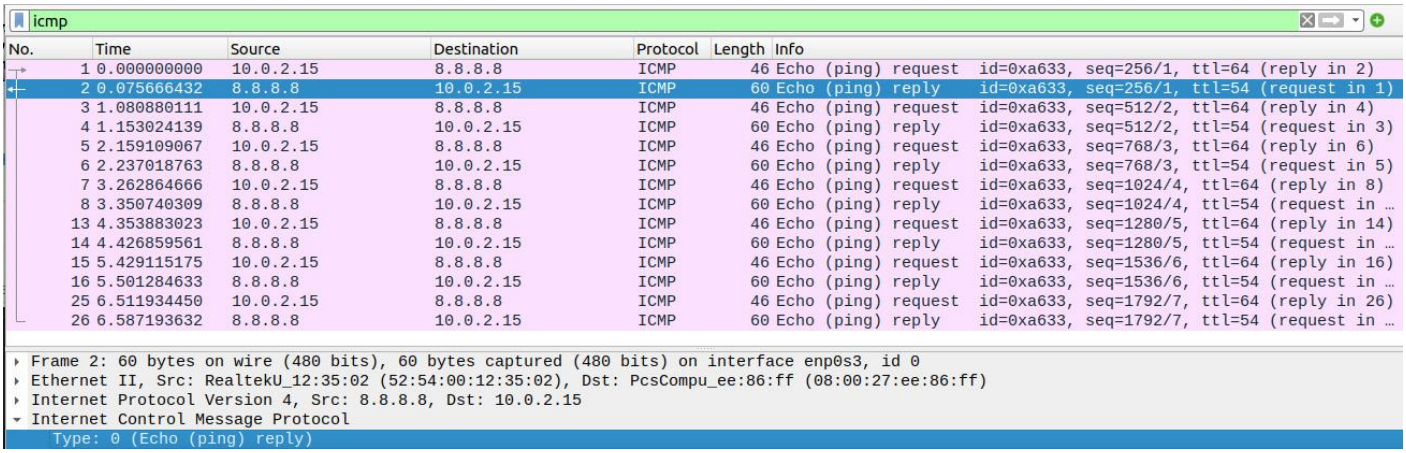
כתובת IP זו אכן קיימת וזמינה, ולכן הודפס פלט בהתאם למבוקש במטלה עבור כל Ping. לאחר זמן קצר עצרנו את הריצה באמצעות Ctrl+c.

להלן תיעוד ההסנפה:



ניתן לראות כי ישנן חבילות בקשה ותשובה. קונקרטית, ניתן לראות בפירוט החבילה של חבילה 1 (מסומנת בכחול) כי הוא מסוג (Type, מסומן בכחול) 8, המעידה על בקשה.

להלן תיעוד נוסף, המעיד על כך שחבילה 2 (מסומנת בכחול) מסוג (Type, מסומן בכחול) 0, המעיד על תשובה:



Alon Meshulam - 207964487
Israel Gitler - 208580076

חלק ב

בחלק זה השתמשנו בשני ממשקים באופן מקביל: enp0s3 (המאפשר לזהות תעבורת ICMP באופן חיצוני) וכן Loopback (העוקב בין היתר אחר תעבורת TCP באופן מקומי). כמו כן, בחלק זה סיננו את ההסנפות לפי הפילטר (icmp || tcp.dstport == 3000 || tcp.srcport == 3000), מאחר שאנו מעוניינים לעקוב הן אחרי פרוטוקול ICMP (בקשות ה-Ping ותשובותיהן) והן אחרי תקשורת ה-TCP עם ה-watchdog.

בפעם הראשונה הרצנו את better_ping.py עם כתובת ה-IP 1.2.3.4 וקיבלנו את הפלט הבא:

```
alon@alonm:~/PycharmProjects/net4$ sudo python3 better_ping.py 1.2.3.4
*****Watchdog: listening on port 3000...*****
Betterping connected to watchdog
Watchdog connected to betterping program
PING 1.2.3.4
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
server 1.2.3.4 cannot be reached.
Watchdog timeout (10 seconds). Close sockets and END.
```

כתובת IP זו אינה קיימת ועל כן לא ציפינו לקבל תשובה (Pong) כאשר הגדרנו בקוד Timeout בן שנייה לקבלת תשובה. לאחר עשר שניות (כמוגדר במטלה) בהן לא נענה אף Ping, מודפסת הודעה מתאימה וה-Watchdog עוצר את התוכנית.

להלן תיעוד החלק העליון של ההסנפה:

(icmp || tcp.dstport==3000 || tcp.srcport == 3000)

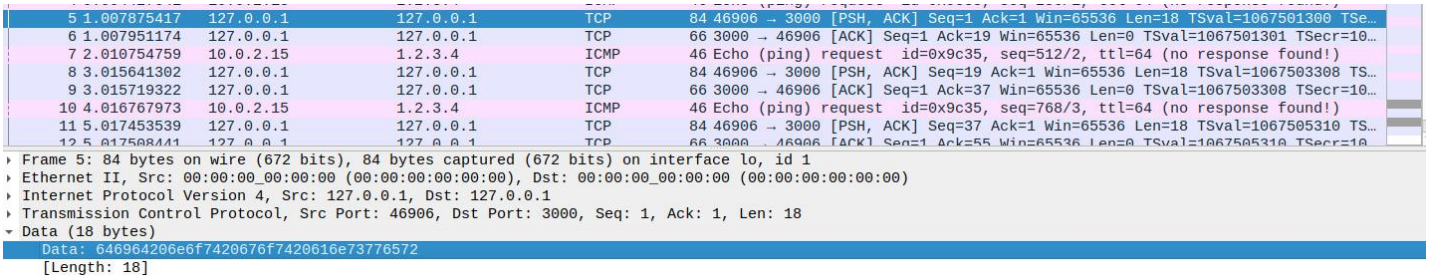
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	127.0.0.1	127.0.0.1	TCP	74	46906 → 3000 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=1067…
2	0.000089721	127.0.0.1	127.0.0.1	TCP	74	3000 → 46906 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1…
3	0.000108808	127.0.0.1	127.0.0.1	TCP	66	46906 → 3000 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=1067500293 TSecr=106…
4	0.004427641	10.0.2.15	1.2.3.4	ICMP	46	Echo (ping) request id=0x9c35, seq=256/1, ttl=64 (no response found!)
5	1.007875417	127.0.0.1	127.0.0.1	TCP	84	46906 → 3000 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=18 TSval=1067501300 TSe…
6	1.007951174	127.0.0.1	127.0.0.1	TCP	66	3000 → 46906 [ACK] Seq=1 Ack=19 Win=65536 Len=0 TSval=1067501301 TSecr=10…
7	2.010754759	10.0.2.15	1.2.3.4	ICMP	46	Echo (ping) request id=0x9c35, seq=512/2, ttl=64 (no response found!)
8	3.015641302	127.0.0.1	127.0.0.1	TCP	84	46906 → 3000 [PSH, ACK] Seq=19 Ack=1 Win=65536 Len=18 TSval=1067503308 TS…
9	3.015719322	127.0.0.1	127.0.0.1	TCP	66	3000 → 46906 [ACK] Seq=1 Ack=37 Win=65536 Len=0 TSval=1067503308 TSecr=10…
10	4.016767973	10.0.2.15	1.2.3.4	ICMP	46	Echo (ping) request id=0x9c35, seq=768/3, ttl=64 (no response found!)
11	5.017453539	127.0.0.1	127.0.0.1	TCP	84	46906 → 3000 [PSH, ACK] Seq=37 Ack=1 Win=65536 Len=18 TSval=1067505310 TS…
12	5.017508441	127.0.0.1	127.0.0.1	TCP	66	3000 → 46906 [ACK] Seq=1 Ack=55 Win=65536 Len=0 TSval=1067505310 TSecr=10…
13	6.022311037	10.0.2.15	1.2.3.4	ICMP	46	Echo (ping) request id=0x9c35, seq=1024/4, ttl=64 (no response found!)
14	7.023596268	127.0.0.1	127.0.0.1	TCP	84	46906 → 3000 [PSH, ACK] Seq=55 Ack=1 Win=65536 Len=18 TSval=1067507316 TS…
15	7.023674724	127.0.0.1	127.0.0.1	TCP	66	3000 → 46906 [ACK] Seq=1 Ack=73 Win=65536 Len=0 TSval=1067507316 TSecr=10…
16	8.029708508	10.0.2.15	1.2.3.4	ICMP	46	Echo (ping) request id=0x9c35, seq=1280/5, ttl=64 (no response found!)
17	9.031355558	127.0.0.1	127.0.0.1	TCP	84	46906 → 3000 [PSH, ACK] Seq=73 Ack=1 Win=65536 Len=18 TSval=1067509324 TS…
18	9.031415859	127.0.0.1	127.0.0.1	TCP	66	3000 → 46906 [ACK] Seq=1 Ack=91 Win=65536 Len=0 TSval=1067509324 TSecr=10…
19	10.032972260	10.0.2.15	1.2.3.4	ICMP	46	Echo (ping) request id=0x9c35, seq=1536/6, ttl=64 (no response found!)
20	11.038570355	127.0.0.1	127.0.0.1	TCP	84	46906 → 3000 [PSH, ACK] Seq=91 Ack=1 Win=65536 Len=18 TSval=1067511331 TS…
21	11.038659819	127.0.0.1	127.0.0.1	TCP	66	3000 → 46906 [ACK] Seq=1 Ack=109 Win=65536 Len=0 TSval=1067511331 TSecr=1…
22	11.039677939	127.0.0.1	127.0.0.1	TCP	66	3000 → 46906 [FIN, ACK] Seq=1 Ack=109 Win=65536 Len=0 TSval=1067511332 TS…
23	11.043729178	127.0.0.1	127.0.0.1	TCP	66	46906 → 3000 [ACK] Seq=109 Ack=2 Win=65536 Len=0 TSval=1067511336 TSecr=1…

ניתן לראות שאכן התבצעה התחברות TCP (SYN, SYN ACK) בין ה-watchdog ל-better_ping (לחיצת יד משולשת), ולאחר מכן ניתן לראות בקשות חוזרות ללא מענה (בהתאם למצופה ולפלט), וכן

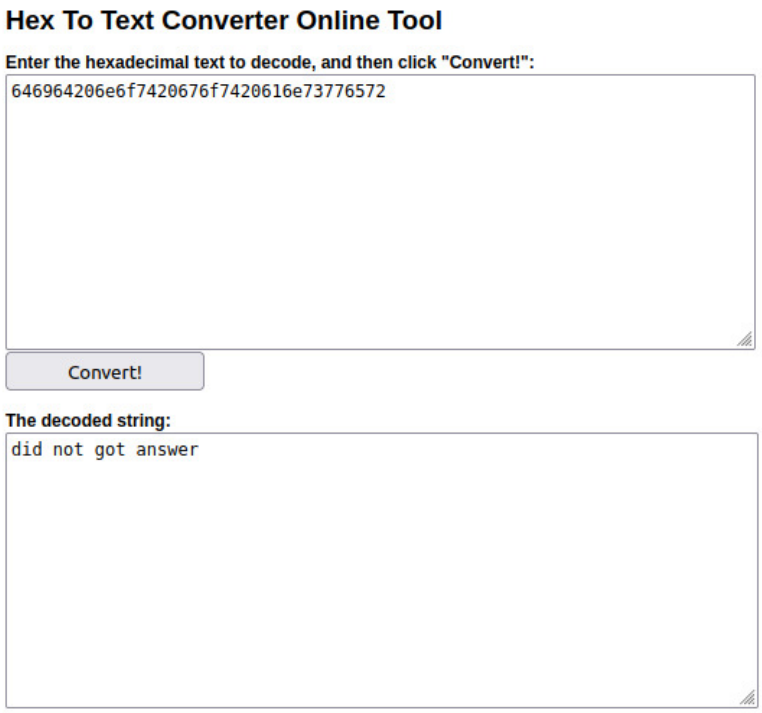
Alon Meshulam - 207964487
Israel Gitler - 208580076

את העדכונים של better_ping ל-watchdog כי טרם התקבלה תשובה (PSH ACK, ACK) בהתאם לקוד שכתבנו.

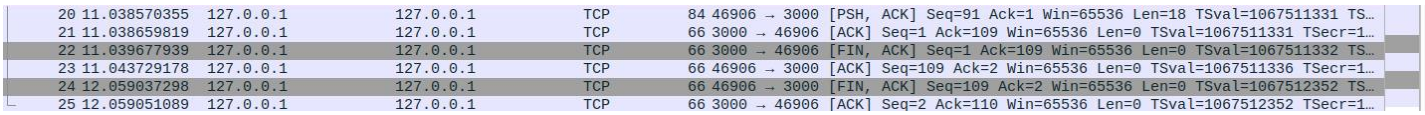
במעבר על כל חבילת PSH ACK נוכל לראות כי היא מכילה שדה DATA הכולל שדה הקסה-דצימלי אותו שלחנו ל-watchdog. להלן דוגמה (החבילה והשדה מסומנים בכחול):



כדי להמיר את תוכן ההודעה נעזרנו באתר הממיר מחרוזת הקסה דצימלית לטקסט (https://string-functions.com/hex-string.aspx) והתוצאה להלן:



להלן תיעוד החלק התחתון של ההסנפה:



בסיום ההסנפה ניתן לראות שהקשרים נסגרו.

Alon Meshulam - 207964487
Israel Gitler - 208580076

בפעם השנייה הרצנו את better_ping.py עם כתובת ה-IP 8.8.8.8 (Google) וקיבלנו את הפלט הבא:

```
alon@alonm:~/PycharmProjects/net4$ sudo python3 better_ping.py 8.8.8.8
*****Watchdog: listening on port 3000...*****
Watchdog connected to betterping program
Betterping connected to watchdog
PING 8.8.8.8
32 bytes from 8.8.8.8 icmp_seq=1 ttl=54 time=73.233 ms
32 bytes from 8.8.8.8 icmp_seq=2 ttl=54 time=736.499 ms
32 bytes from 8.8.8.8 icmp_seq=3 ttl=54 time=87.541 ms
32 bytes from 8.8.8.8 icmp_seq=4 ttl=54 time=73.556 ms
32 bytes from 8.8.8.8 icmp_seq=5 ttl=54 time=73.018 ms
32 bytes from 8.8.8.8 icmp_seq=6 ttl=54 time=70.778 ms
32 bytes from 8.8.8.8 icmp_seq=7 ttl=54 time=71.725 ms
32 bytes from 8.8.8.8 icmp_seq=8 ttl=54 time=450.633 ms
32 bytes from 8.8.8.8 icmp_seq=9 ttl=54 time=76.024 ms
32 bytes from 8.8.8.8 icmp_seq=10 ttl=54 time=74.128 ms
^CPing stopped by Ctrl-c
Close watchdog
```

כתובת IP זו אכן קיימת וזמינה, ולכן הודפס פלט בהתאם למבוקש במטלה עבור כל Ping. לאחר זמן קצר עצרנו את הריצה באמצעות Ctrl+c.

להלן תיעוד של החלק העליון של ההסנפה:

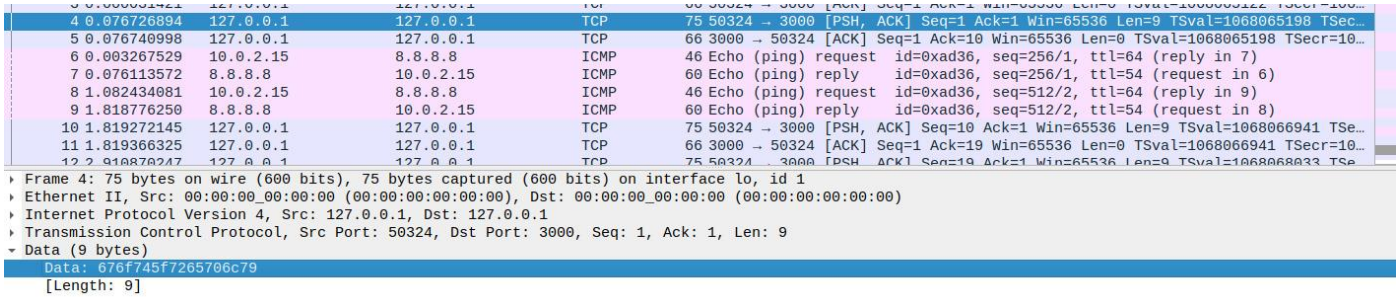
(icmp || tcp.dstport==3000 || tcp.srcport == 3000)

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	127.0.0.1	127.0.0.1	TCP	74	50324 → 3000 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=1068…
2	0.000022418	127.0.0.1	127.0.0.1	TCP	74	3000 → 50324 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1…
3	0.000031421	127.0.0.1	127.0.0.1	TCP	66	50324 → 3000 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=1068065122 TSecr=106…
4	0.076726894	127.0.0.1	127.0.0.1	TCP	75	50324 → 3000 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=9 TSval=1068065198 TSec…
5	0.076740998	127.0.0.1	127.0.0.1	TCP	66	3000 → 50324 [ACK] Seq=1 Ack=10 Win=65536 Len=0 TSval=1068065198 TSecr=10…
6	0.003267529	10.0.2.15	8.8.8.8	ICMP	46	Echo (ping) request id=0xad36, seq=256/1, ttl=64 (reply in 7)
7	0.076113572	8.8.8.8	10.0.2.15	ICMP	60	Echo (ping) reply id=0xad36, seq=256/1, ttl=54 (request in 6)
8	1.082434081	10.0.2.15	8.8.8.8	ICMP	46	Echo (ping) request id=0xad36, seq=512/2, ttl=64 (reply in 9)
9	1.818776250	8.8.8.8	10.0.2.15	ICMP	60	Echo (ping) reply id=0xad36, seq=512/2, ttl=54 (request in 8)
10	1.819272145	127.0.0.1	127.0.0.1	TCP	75	50324 → 3000 [PSH, ACK] Seq=10 Ack=1 Win=65536 Len=9 TSval=1068066941 TSe…
11	1.819366325	127.0.0.1	127.0.0.1	TCP	66	3000 → 50324 [ACK] Seq=1 Ack=19 Win=65536 Len=0 TSval=1068066941 TSecr=10…
12	2.910870247	127.0.0.1	127.0.0.1	TCP	75	50324 → 3000 [PSH, ACK] Seq=19 Ack=1 Win=65536 Len=9 TSval=1068068033 TSe…
13	2.910960358	127.0.0.1	127.0.0.1	TCP	66	3000 → 50324 [ACK] Seq=1 Ack=28 Win=65536 Len=0 TSval=1068068033 TSecr=10…
14	2.823188038	10.0.2.15	8.8.8.8	ICMP	46	Echo (ping) request id=0xad36, seq=768/3, ttl=64 (reply in 15)
15	2.908958280	8.8.8.8	10.0.2.15	ICMP	60	Echo (ping) reply id=0xad36, seq=768/3, ttl=54 (request in 14)
16	3.912848884	10.0.2.15	8.8.8.8	ICMP	46	Echo (ping) request id=0xad36, seq=1024/4, ttl=64 (reply in 17)
17	3.986153109	8.8.8.8	10.0.2.15	ICMP	60	Echo (ping) reply id=0xad36, seq=1024/4, ttl=54 (request in 16)
18	3.986532041	127.0.0.1	127.0.0.1	TCP	75	50324 → 3000 [PSH, ACK] Seq=28 Ack=1 Win=65536 Len=9 TSval=1068069108 TSe…
19	3.986780663	127.0.0.1	127.0.0.1	TCP	66	3000 → 50324 [ACK] Seq=1 Ack=37 Win=65536 Len=0 TSval=1068069108 TSecr=10…
20	5.062718047	127.0.0.1	127.0.0.1	TCP	75	50324 → 3000 [PSH, ACK] Seq=37 Ack=1 Win=65536 Len=9 TSval=1068070184 TSe…
21	5.062783585	127.0.0.1	127.0.0.1	TCP	66	3000 → 50324 [ACK] Seq=1 Ack=46 Win=65536 Len=0 TSval=1068070184 TSecr=10…
22	4.989587424	10.0.2.15	8.8.8.8	ICMP	46	Echo (ping) request id=0xad36, seq=1280/5, ttl=64 (reply in 23)
23	5.062276613	8.8.8.8	10.0.2.15	ICMP	60	Echo (ping) reply id=0xad36, seq=1280/5, ttl=54 (request in 22)
24	6.064007182	10.0.2.15	8.8.8.8	ICMP	46	Echo (ping) request id=0xad36, seq=1536/6, ttl=64 (reply in 25)

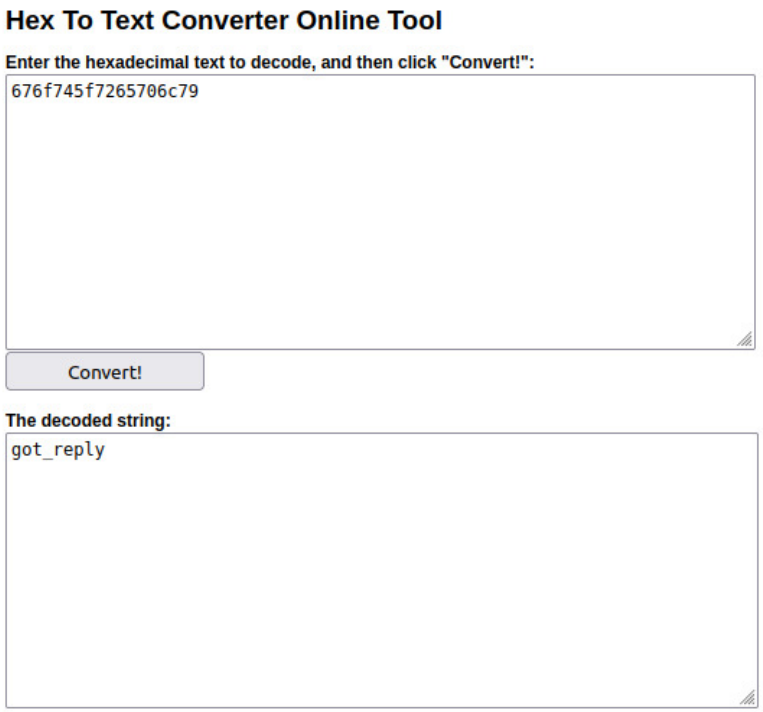
ניתן לראות שאכן התבצעה התחברות TCP (SYN, SYN ACK) בין ה-watchdog ל-better_ping (לחיצת יד משולשת), ולאחר מכן ניתן לראות בקשות (Pings) ותשובות (בהתאם למצופה ולפלט). בין לבין ניתן לראות את העדכונים של better_ping ל-watchdog כי התקבלה תשובה (PSH ACK, ACK) בהתאם לקוד שכתבנו. במעבר על כל חבילת PSH ACK נוכל לראות כי היא מכילה שדה

Alon Meshulam - 207964487
Israel Gitler - 208580076

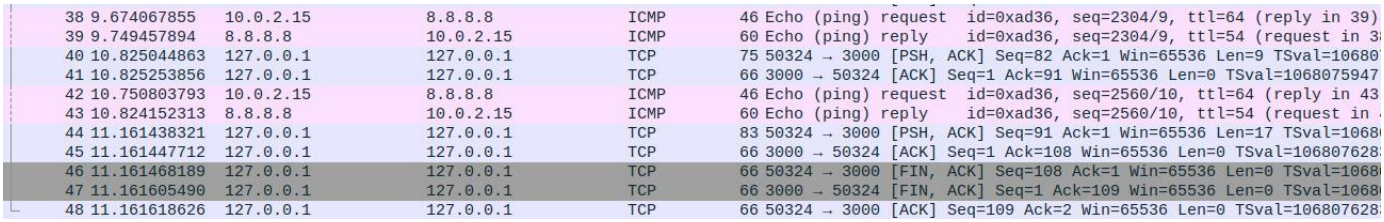
DATA הכולל שדה הקסה-דצימלי אותו שלחנו ל-watchdog. להלן דוגמה (החבילה והשדה מסומנים בכחול):



כדי להמיר את תוכן ההודעה נעזרנו באתר הממיר מחרוזת הקסה דצימלית לטקסט (https://string-functions.com/hex-string.aspx) והתוצאה להלן:



להלן תיעוד של החלק התחתון של ההסנפה:



בסיום ההסנפה ניתן לראות שהקשרים נסגרו.