

# תרחיש 3 : בניית מתקפת DDOS

## 1. לוגיסטיקה :

שם מגישים : רון אמסלם  
פאדי נוג'ידאת

## 2. תהליך בניית המתקפה :

במתקפה זו ביצענו סימולציה של מתקפת SYN Flood בה התוקף שולח כמות עצומה של בקשות SYN לשרת במטרה להעמיס עליו ולגרום לו לא להיות זמין. במקביל, שלחנו פינגים מהמוניטור אל השרת כדי לבדוק האם זמני התגובה של השרת נפגעים במהלך המתקפה.

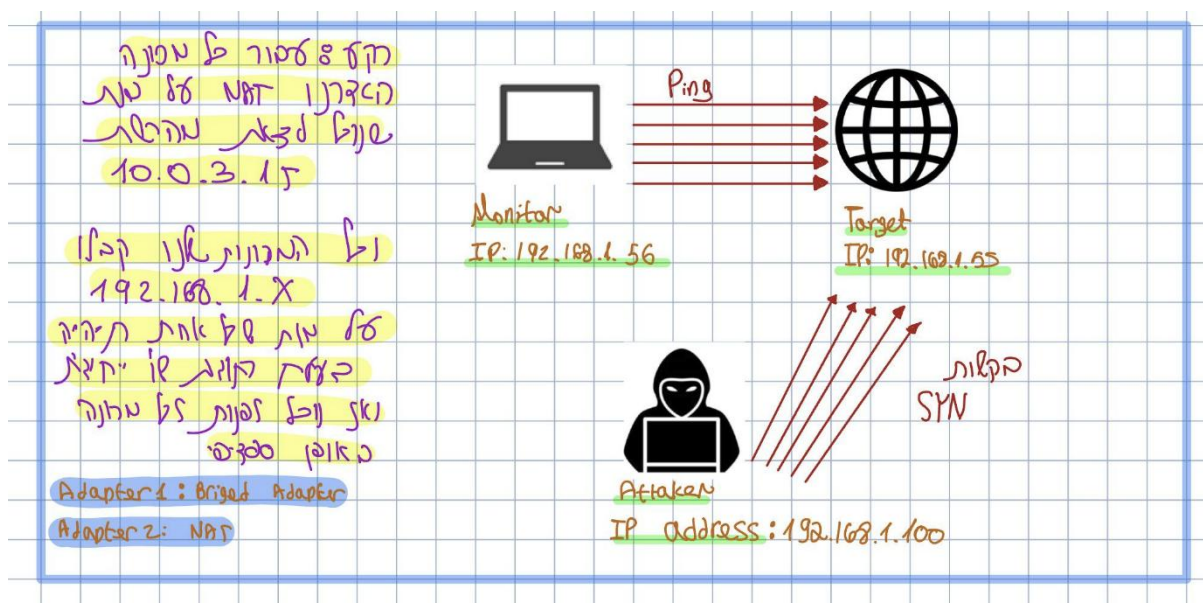
ראשית נדון בבניית המכונות שלנו

Attacker.1

Monitor.2

Target.3

להלן תרחיש שמתאר את המתרחש במתקפה שלנו הכולל כתובת IP עבור כל אחת מהמכונות .



במתקפה , אנחנו בודקים את ההשפעה על קצב שרת web , apache2. מה שבעצם קורה זה שאנחנו שולחים הודעות SYN מהתוקף אל שרת web ויש לנו Monitor שמאזין ושולח פינג אל השרת שלנו , ואנחנו בעצם בודקים את העיכוב בזמן בגעת הפינג אל המוניטור ובכך מודדים את ההשפעה על השרת.

בתרחיש זה התעסקנו בלקודד שתי תוכניות אחת בשפת פיתון והשנייה בשפת C וחלק ממטרת התרחיש היא להשוואת בין שתי התוכניות בפרמטרים שונים כגון : סטיית תקן , ממוצע , זמני RTT ולהדגיש את ההבדלים בין שימוש בשפת C לשימוש בשפת פיתון . כמו כן התבקשנו לשמור את כל המידע בקובצים על מנת שלאחר מכן נושה בין התוצאות שקיבלנו על ידי יצירת גרפים .

בסביבת העבודה השתמשנו ב־virtual box והתקנו שתי מכונות שונות על מנת למנוע תקלות טכניות

Ubuntu 23.04 Lunar Lobster



Ubuntu 24.10 Oracular Oriole

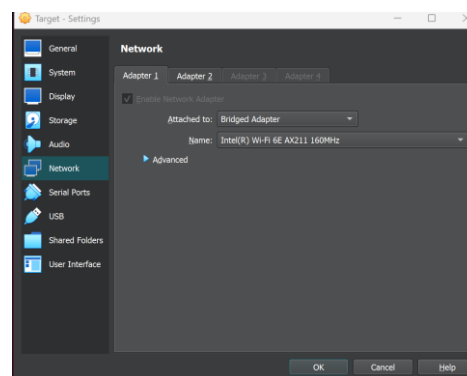
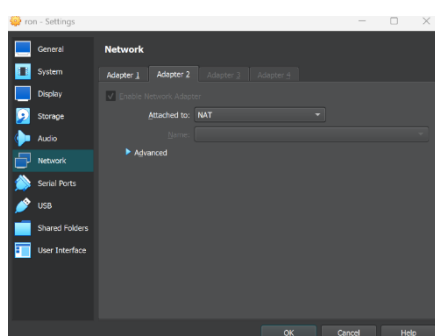


עבור המכונה השלישית השתמשנו במה שהיה מותקן כבר אובונטו 24.10.

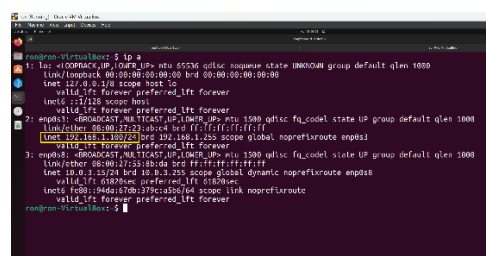
על מנת ליצור סביבת עבודה מספקת מה שעשינו זה הגדרנו עבור כל מכונה את ה Network באופן הבא השתמשנו ב NAT על מנת שנוכל לתקשר עם האינטרנט החיצוני (נדרש על מנת לגשת לשרת ה web שלנו ).

במהלך המתקפה אנחנו משתמשים בשליחת חבילת SYN מצד התוקף , ובעצם אנחנו מנצלים את מנגנון פתיחת הקשר של פרוטוקול TCP (שלב הראשון הוא שליחת SYN ) , מהסיבה ש TCP הוא פרוטוקול אמין אז הוא מגיב לחבילה , והשרת מקצה משאבים להתחברות הלקוח , אבל אנחנו לא ממשיכים להתחברות לשרת ולכן החיבורים הנוצרים הם חיבורים הנמצאים בחצי חיבור פתוח, ואנחנו מבצעים את זה מליון פעמים כך שבסופו של דבר אנחנו גורמים לעומס רב על השרת ולתפקוד איטי ואף לקוי של השרת עבור לקוחות חדשים או לקוחות שכבר נמצאים בשירות .

השתמשנו גם ב Bridged על מנת שכל שלושת המכונות יקבלו IP ייחודי בכדי שנוכל להבדיל בניהם ויכלו לתקשר אחת עם השנייה כמו שמופיע למטה :



תוקף :



## מוניטור:

```
osboxes@osboxes:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp8s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:5a:0b:68 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.56/24 brd 192.168.1.255 scope global dynamic noprefixroute enp8s3
        valid_lft 962sec preferred_lft 962sec
    inet6 fe80::de47:ff90:47d:7a4/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
3: enp8s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:5a:0b:68 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.57/24 brd 192.168.1.255 scope global dynamic noprefixroute enp8s8
        valid_lft 6189sec preferred_lft 6189sec
    inet6 fe80::5ff:908e:825b:f163/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
osboxes@osboxes:~$
```

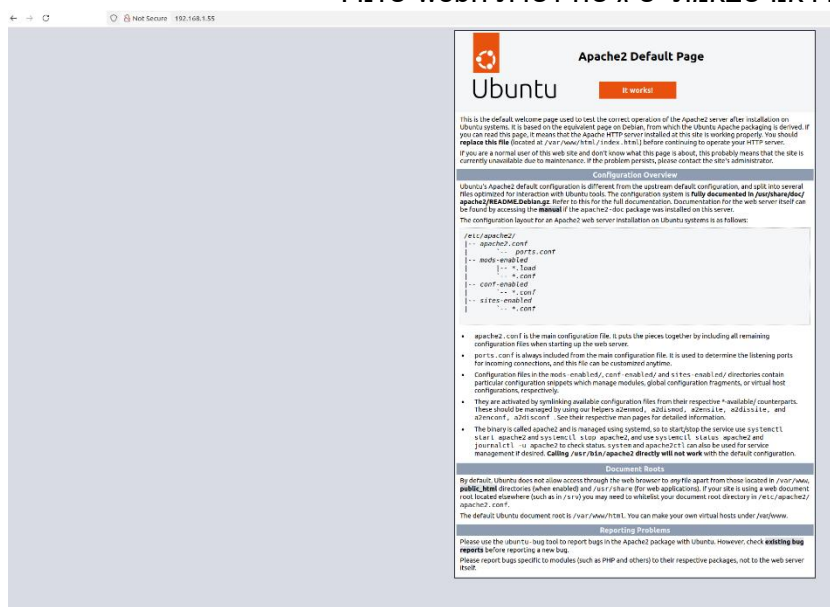
יעד :

```
osboxes@osboxes:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: enp8s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:5a:0b:68 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.57/24 brd 192.168.1.255 scope global dynamic noprefixroute enp8s3
        valid_lft 3033sec preferred_lft 3033sec
    inet6 fe80::55a4:1b05:d697:22a7/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
3: enp8s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:5a:0b:68 brd ff:ff:ff:ff:ff:ff
osboxes@osboxes:~$
```

כמו כן הפעלנו את השירות של apache2 במכונת היעד שלנו:

```
osboxes@osboxes:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: enabled)
   Active: active (running) since Thu 2025-04-10 10:09:33 EDT; 6h ago
     Invocation: 17a3565d2c324cd18b334b1c3182940c
       Docs: https://httpd.apache.org/docs/2.4/
    Process: 1348 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
```

וידאנו שבאמת יש גישה לשרת הweb שלנו :



במהלך המעבדה הרצנו מתקפת SYN פעמיים – פעם אחת באמצעות קוד בשפת C ופעם נוספת באמצעות קוד בפיתון. בכל אחת מההרצות מדדנו את זמני השליחה של בקשות ה-SYN ואת זמני התגובה של השרת, בעזרת פינגים שנשלחו מהמוניטור.

אנחנו מניחים שיכולה להיות הבדל בביצועים בין שתי השפות – כאשר C עשויה להיות מהירה כי היא יעילה יותר מהסיבה שהיא יותר קרובה לשפת מכונה, ופיתון איטית יותר כי יש לה את תהליך התרגום שלוקח הרבה יותר זמן מ-C.

## מתקפת SYN באמצעות קידוד בשפת C

בצד התוקף יצרנו קובץ בסיומת C בשם attack1.c בעזרת הכלי scanh ערכנו וכתבנו קוד אשר שולח בקשות SYN לשרת כנדרש. עשינו 100 איטרציות כך שבכל איטרציה נשלחות 10,000 פאקטות.

במימוש התוקף מתחזה לכתובת 192.168.1.2 ושולח הודעות ממנו.

בצד המוניטור יצרנו קובץ בסיומת C בשם ping4.c בעזרת הכלי scanh ערכנו וכתבנו קוד אשר שולח בקשות ping לשרת כל 5 שניות.

קמפלנו את הקבצים שלנו בעזרת gcc

```
ron@ron-VirtualBox:~/Desktop$ nano attack1.c
ron@ron-VirtualBox:~/Desktop$ gcc attack1.c -o attack1 -lm
```

```
osboxes@osboxes:~$ nano ping4.c
osboxes@osboxes:~$ gcc ping4.c -o ping4 -lm
```

כמו כן התקנו במכונה של ה Target Wireshark על מנת לבדוק את התעבורה במהלך המתקפה.

כאן ניתן לראות את הבקשות SYN מצד התוקף אל כיוון השרת:

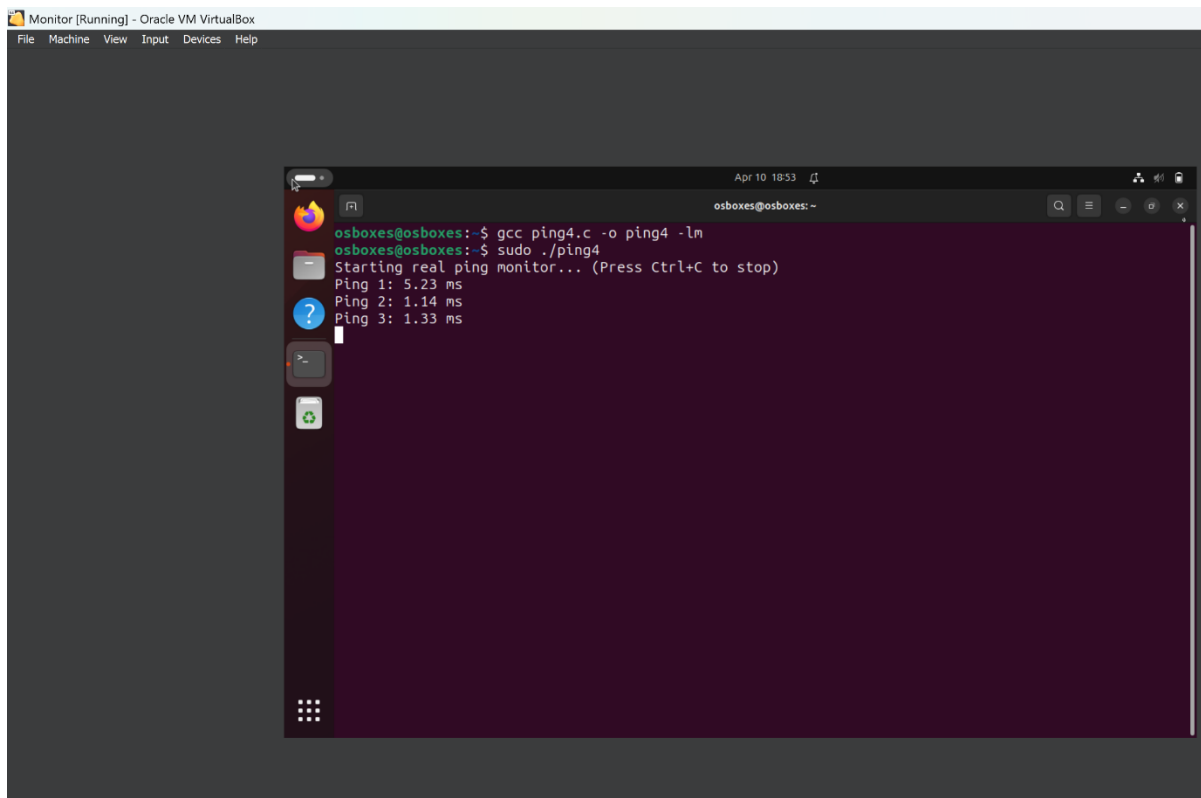
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
2	0.000440395	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
3	0.000440436	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
4	0.000821698	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
5	0.000821634	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
6	0.001340084	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
7	0.001519031	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
8	0.001519076	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
9	0.001976692	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
10	0.001976751	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
11	0.001976796	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
12	0.002387563	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
13	0.002387599	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
14	0.002387631	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
15	0.002387663	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
16	0.002387695	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
17	0.002387727	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
18	0.002387759	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
19	0.002387791	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
20	0.002387823	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
21	0.002387855	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
22	0.002387887	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
23	0.002387919	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
24	0.002387951	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
25	0.002387983	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
26	0.002388015	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
27	0.002388047	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
28	0.002388079	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
29	0.002388111	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
30	0.002388143	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
31	0.002388175	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
32	0.002388207	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0
33	0.002388239	192.168.1.2	192.168.1.55	TCP	60	TCP [Retransmission] 1234 -> 60 [SYN] Seq=1234, Win=0

בצד המוניטור כתבנו קוד אשר ישלח בקשות PING לשרת על מנת שנוכל למדוד את העומס

No.	Time	Source	Destination	Protocol	Length	Info
8337	2.020932146	192.168.1.56	192.168.1.55	ICMP	62	Echo (ping) request id=0xe01a, seq=4096/16, ttl=64 (r)
8340	2.020954457	192.168.1.55	192.168.1.56	ICMP	62	Echo (ping) reply id=0xe01a, seq=4096/16, ttl=64 (r)
25495	7.028855473	192.168.1.56	192.168.1.55	ICMP	62	Echo (ping) request id=0xe01a, seq=4352/17, ttl=64 (r)
25497	7.028873133	192.168.1.55	192.168.1.56	ICMP	62	Echo (ping) reply id=0xe01a, seq=4352/17, ttl=64 (r)
35044	12.035096579	192.168.1.56	192.168.1.55	ICMP	62	Echo (ping) request id=0xe01a, seq=4608/18, ttl=64 (r)
35045	12.035121608	192.168.1.55	192.168.1.56	ICMP	62	Echo (ping) reply id=0xe01a, seq=4608/18, ttl=64 (r)
44752	17.040150124	192.168.1.56	192.168.1.55	ICMP	62	Echo (ping) request id=0xe01a, seq=4864/19, ttl=64 (r)
44756	17.040173969	192.168.1.55	192.168.1.56	ICMP	62	Echo (ping) reply id=0xe01a, seq=4864/19, ttl=64 (r)
54527	22.045425736	192.168.1.56	192.168.1.55	ICMP	62	Echo (ping) request id=0xe01a, seq=5120/20, ttl=64 (r)
54528	22.045443912	192.168.1.55	192.168.1.56	ICMP	62	Echo (ping) reply id=0xe01a, seq=5120/20, ttl=64 (r)
64825	27.058342264	192.168.1.56	192.168.1.55	ICMP	62	Echo (ping) request id=0xe01a, seq=5376/21, ttl=64 (r)
64833	27.058367726	192.168.1.55	192.168.1.56	ICMP	62	Echo (ping) reply id=0xe01a, seq=5376/21, ttl=64 (r)
75058	32.063252726	192.168.1.56	192.168.1.55	ICMP	62	Echo (ping) request id=0xe01a, seq=5632/22, ttl=64 (r)
75059	32.063277439	192.168.1.55	192.168.1.56	ICMP	62	Echo (ping) reply id=0xe01a, seq=5632/22, ttl=64 (r)

(השעות בין המכונות שונות)

התחלנו ראשית לשלוח ping מהמוניטור שלנו אל השרת 10.4.25 בשעה 18:53



ואת המתקפה התחלנו מיד אחרי שנשלחו כמה פינגים בשעה 17:43 10.4.25

```
ron@ron-VirtualBox: ~/Desktop$ sudo ./attack1
Starting attack...
Sent 10000 packets
Sent 20000 packets
Sent 30000 packets
Sent 40000 packets
Sent 50000 packets
```

המתקפה הסתיימה אחרי 5 דקות ו34 שניות

ונוצרו לנו קבצים הבאים

מהמוניטור : pings\_results\_c.txt בפורמט הבא :

```
1 Ping 1: 1.361000 ms
2 Ping 2: 1.237000 ms
3 Ping 3: 1.328000 ms
4 Ping 4: 0.935000 ms
5 Ping 5: 0.869000 ms
6 Ping 6: 2.863000 ms
7 Ping 7: 4.213000 ms
8 Ping 8: 1.070000 ms
9 Ping 9: 0.784000 ms
10 Ping 10: 0.786000 ms
11 Ping 11: 0.764000 ms
12 Ping 12: 0.838000 ms
13 Ping 13: 1.258000 ms
14 Ping 14: 0.378000 ms
15 Ping 15: 0.842000 ms
16 Ping 16: 6.675000 ms
17 Ping 17: 6.921000 ms
18 Ping 18: 1.325000 ms
19 Ping 19: 0.870000 ms
20 Ping 20: 3.325000 ms
21 Ping 21: 0.692000 ms
```

ובסופו ממוצע וסטיית תקן :

```

77 Ping 77: 1.696000 ms
78 Ping 78: 0.997000 ms
79
80 Average RTT: 1.640167 ms
81 Standard Deviation: 1.472180 ms
82

```

מהתוקף : syns\_results\_c.txt בפורמט הבא :

```

Packet 1: 0.134221 ms
Packet 2: 0.040636 ms
Packet 3: 0.030584 ms
Packet 4: 0.028601 ms
Packet 5: 0.028188 ms
Packet 6: 0.025639 ms
Packet 7: 0.029348 ms
Packet 8: 0.028133 ms
Packet 9: 0.027669 ms
Packet 10: 0.028441 ms
Packet 11: 0.030434 ms
Packet 12: 0.149167 ms
Packet 13: 0.071998 ms
Packet 14: 0.045793 ms
Packet 15: 0.029007 ms
Packet 16: 0.026618 ms
Packet 17: 0.026536 ms
Packet 18: 0.025767 ms
Packet 19: 0.025657 ms
Packet 20: 0.028334 ms
Packet 21: 0.027216 ms

```

בסופו זמן הכולל, ממוצע, וסטיית תקן

```

Packet 777777: 0.140211 ms
Packet 1000000: 0.140483 ms

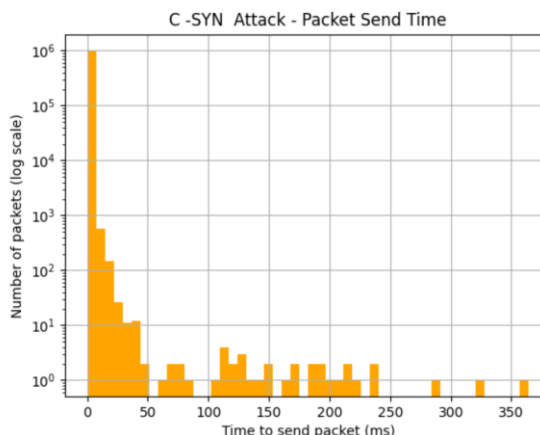
Total time: 373.05 seconds
Average time: 0.372858 ms
Standard Deviation: 1.183974 ms

```

מהנתונים הסטטיסטיים עולה כי הזמן הממוצע לשליחת חבילת SYN עמד על כ-0.37 מילישניות בלבד, דבר המעיד על קצב שליחה גבוה מאוד. עם זאת, סטיית התקן שעמדה על כ-1.18 מילישניות מצביעה על כך שלא כל הבקשות נשלחו באותו הקצב – חלקן לקחו משמעותית יותר זמן. פיזור זה מתבטא גם בגרף ההיסטוגרמה, בו ניתן לראות בקשות בודדות שהתעכבו.



## גרף ראשון המבטא את - זמן שליחה של בקשות SYN



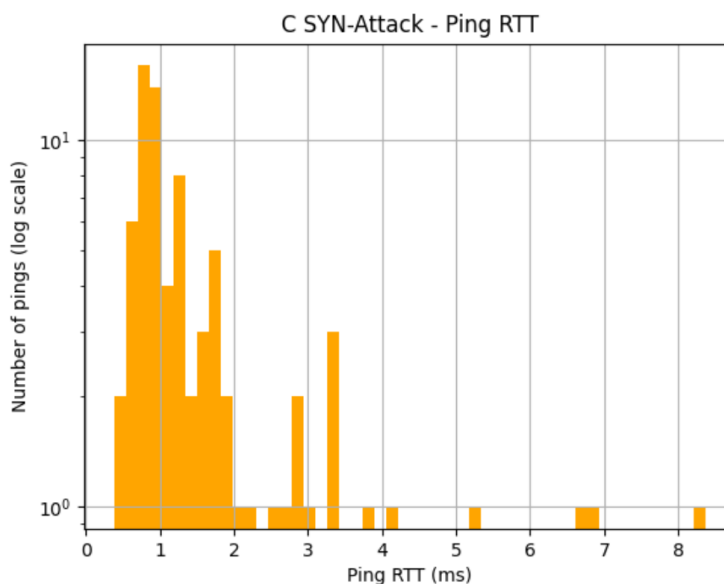
הגרף מציג כמה זמן לקח לשלוח כל חבילת SYN מהתוקף אל השרת.

בציר X: הזמן לשליחת חבילה אחת (במילישניות)

בציר Y: מספר הפאקטות שנשלחו בטווח הזמן הזה (בסקלה לוגריתמית).

ההתפלגות מראה שרוב החבילות נשלחו מהר, עם שיא גדול סביב 0–5 ms. אבל יש גם זנב ארוך של חבילות שנשלחו בצורה איטית יותר – חלק אפילו מעל 300 ms. זה כנראה קרה בגלל עומס נקודתי במערכת, שגרם לעיכובים קטנים.

## גרף שני המבטא את זמן תגובה של השרת (RTT)



הגרף מציג את ה-RTT של הפינגים ששלח המוניטור במהלך התקיפה.

בציר X: זמן תגובה של פינג – במילישניות.

בציר Y: כמה פינגים התקבלו עם זמן כזה (בסקלה לוגריתמית).

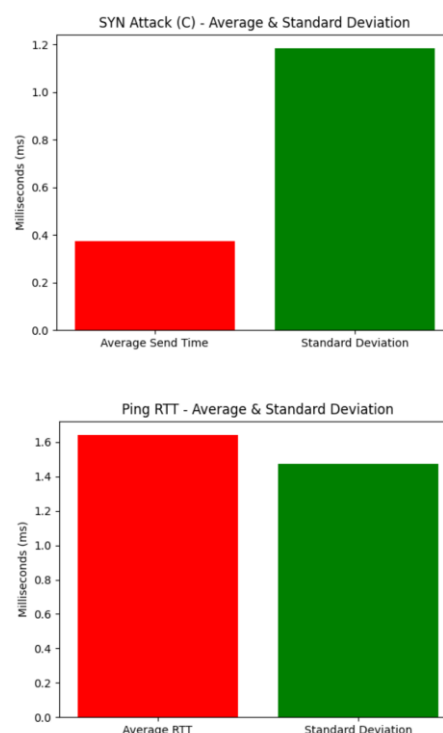
רוב הפינגים התקבלו ב-RTT נמוך כלומר השרת מגיב מהר.

חלק קטן מהפינגים לקח להם יותר זמן.

אפשר לראות שההתפלגות מאוד מרוכזת בין 0.5 ל-2.5 מילישניות, עם שיא סביב 1 מילישניות כלומר הרוב המוחלט של הפינגים קיבלו תגובה ממש מהירה. יש כמה ערכים בודדים שמגיעים גם ל-4 עד 8 מילישניות אבל הם נדירים ולא משפיעים הרבה. בסה"כ נראה שהשרת הגיב בצורה מאוד יציבה ומהירה לאורך כל התקופה.

מהגרפים אפשר לראות שרוב החבילות נשלחו ממש מהר, כמעט כולן תוך פחות מ-5 מילישניות. גם הפינגים שקיבלנו מהמוניטור בזמן ההתקפה נשארו עם זמן תגובה נמוך ויציב – כמעט כולם בין 0.5 ל-2.5 ms זה אומר שהעומס שהמתקפה יצרה על השרת לא השפיע כמעט בכלל על היכולת שלו לענות לפינגים. למרות ששלחנו מיליון חבילות, נראה שהשליחה הייתה מספיק יציבה ומהירה כדי לא להעמיס יותר מדי על המערכת.

בהמשך לנתונים הסטטיסטים :



הגרף הראשון ממחיש את הפער בין הזמן הממוצע לשליחת חבילת SYN לבין סטיית התקן של זמני השליחה.

ניתן לראות שהממוצע נמוך מאוד, אך סטיית התקן גבוהה יחסית כ-1.18, מה שמעיד על פיזור רחב בזמני השליחה. כלומר, בזמן שרוב הבקשות נשלחו במהירות, חלק קטן מהן התעכב.

בגרף של הפינגים – הממוצע יחסית נמוך, אבל סטיית התקן כמעט זהה אליו. זה מראה שההתקפה לא השביתה את השרת, אבל כן הייתה קצת השפעה לא יציבה על התגובה לפינגים.



## מתקפת SYN באמצעות פיתון

בצד התוקף יצרנו קובץ בסיומת `pc` בשם `attack5.py` בעזרת הכלי `snm` ערכנו וכתבנו קוד אשר שולח בקשות SYN לשרת כנדרש עשינו 100 איטרציות כך שבכל איטרציה נשלחות 10,000 פאקטות .

בצד המוניטור יצרנו קובץ בסיומת `pc` בשם `ping6.c` בעזרת הכלי `snm` ערכנו וכתבנו קוד אשר שולח בקשות ping לשרת כל 5 שניות .

ניתן לראות באותה `wireshark` הנמצא במכונה של היעד , עקב בקשות התוקף :

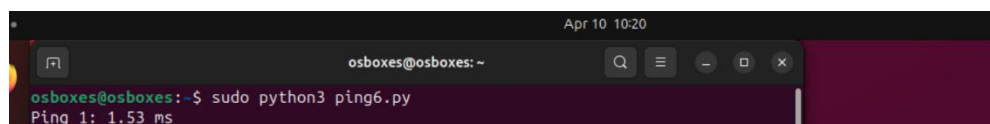
No.	Time	Source	Destination	Protocol	Length	Info
2591	32.024308388	192.168.1.100	192.168.1.55	TCP	60	[TCP Port numbers reused] 29 → 80 [SYN] Seq=0 Win=0 Len=0
2593	32.029037219	192.168.1.100	192.168.1.55	TCP	60	29 → 80 [RST] Seq=1 Win=0 Len=0
2594	32.063887707	192.168.1.100	192.168.1.55	TCP	60	[TCP Port numbers reused] 29 → 80 [SYN] Seq=0 Win=0 Len=0
2596	32.065235572	192.168.1.100	192.168.1.55	TCP	60	29 → 80 [RST] Seq=1 Win=0 Len=0
2597	32.100599614	192.168.1.100	192.168.1.55	TCP	60	[TCP Port numbers reused] 29 → 80 [SYN] Seq=0 Win=0 Len=0
2599	32.114784551	192.168.1.100	192.168.1.55	TCP	60	29 → 80 [RST] Seq=1 Win=0 Len=0
2600	32.120605250	192.168.1.100	192.168.1.55	TCP	60	[TCP Port numbers reused] 29 → 80 [SYN] Seq=0 Win=0 Len=0
2607	32.131458980	192.168.1.100	192.168.1.55	TCP	60	29 → 80 [RST] Seq=1 Win=0 Len=0
2608	32.155027460	192.168.1.100	192.168.1.55	TCP	60	[TCP Port numbers reused] 29 → 80 [SYN] Seq=0 Win=0 Len=0
2610	32.158020520	192.168.1.100	192.168.1.55	TCP	60	29 → 80 [RST] Seq=1 Win=0 Len=0
2611	32.185223749	192.168.1.100	192.168.1.55	TCP	60	[TCP Port numbers reused] 29 → 80 [SYN] Seq=0 Win=0 Len=0
2613	32.186179832	192.168.1.100	192.168.1.55	TCP	60	29 → 80 [RST] Seq=1 Win=0 Len=0
2614	32.205509228	192.168.1.100	192.168.1.55	TCP	60	[TCP Port numbers reused] 29 → 80 [SYN] Seq=0 Win=0 Len=0
2616	32.207352790	192.168.1.100	192.168.1.55	TCP	60	29 → 80 [RST] Seq=1 Win=0 Len=0
2621	32.232056657	192.168.1.100	192.168.1.55	TCP	60	[TCP Port numbers reused] 29 → 80 [SYN] Seq=0 Win=0 Len=0
2623	32.234150943	192.168.1.100	192.168.1.55	TCP	60	29 → 80 [RST] Seq=1 Win=0 Len=0
2624	32.257843436	192.168.1.100	192.168.1.55	TCP	60	[TCP Port numbers reused] 29 → 80 [SYN] Seq=0 Win=0 Len=0
2626	32.260029707	192.168.1.100	192.168.1.55	TCP	60	29 → 80 [RST] Seq=1 Win=0 Len=0
2627	32.282842307	192.168.1.100	192.168.1.55	TCP	60	[TCP Port numbers reused] 29 → 80 [SYN] Seq=0 Win=0 Len=0
2629	32.293747615	192.168.1.100	192.168.1.55	TCP	60	29 → 80 [RST] Seq=1 Win=0 Len=0
2630	32.324912114	192.168.1.100	192.168.1.55	TCP	60	[TCP Port numbers reused] 29 → 80 [SYN] Seq=0 Win=0 Len=0
2632	32.325598102	192.168.1.100	192.168.1.55	TCP	60	29 → 80 [RST] Seq=1 Win=0 Len=0
2633	32.375152167	192.168.1.100	192.168.1.55	TCP	60	[TCP Port numbers reused] 29 → 80 [SYN] Seq=0 Win=0 Len=0
2635	32.376963285	192.168.1.100	192.168.1.55	TCP	60	29 → 80 [RST] Seq=1 Win=0 Len=0
2636	32.419332831	192.168.1.100	192.168.1.55	TCP	60	[TCP Port numbers reused] 29 → 80 [SYN] Seq=0 Win=0 Len=0
2638	32.421168737	192.168.1.100	192.168.1.55	TCP	60	29 → 80 [RST] Seq=1 Win=0 Len=0
2639	32.465092086	192.168.1.100	192.168.1.55	TCP	60	[TCP Port numbers reused] 29 → 80 [SYN] Seq=0 Win=0 Len=0
2641	32.466199772	192.168.1.100	192.168.1.55	TCP	60	29 → 80 [RST] Seq=1 Win=0 Len=0
2642	32.494886831	192.168.1.100	192.168.1.55	TCP	60	[TCP Port numbers reused] 29 → 80 [SYN] Seq=0 Win=0 Len=0
2644	32.496702345	192.168.1.100	192.168.1.55	TCP	60	29 → 80 [RST] Seq=1 Win=0 Len=0
2645	32.517811625	192.168.1.100	192.168.1.55	TCP	60	[TCP Port numbers reused] 29 → 80 [SYN] Seq=0 Win=0 Len=0
2647	32.518703879	192.168.1.100	192.168.1.55	TCP	60	29 → 80 [RST] Seq=1 Win=0 Len=0

כפי שניתן לראות, כמעט כל החבילות הן בקשות SYN בפרוטוקול TCP הנשלחות אל השרת , ניתן להבין מזה שמתקיימת מתקפת SYN Flood שבה התוקף שולח כמויות גדולות של בקשות פתיחת חיבור מכלי להשלים את התהליך.

ובקשות המוניטור:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.1.56	192.168.1.55	ICMP	98	Echo (ping) request id=0x043f, seq=1/256, ttl=64 (rep
8	5.034890340	192.168.1.56	192.168.1.55	ICMP	98	Echo (ping) request id=0xbae1, seq=1/256, ttl=64 (rep
13	10.550219173	192.168.1.56	192.168.1.55	ICMP	98	Echo (ping) request id=0x3993, seq=1/256, ttl=64 (rep
15	15.559576546	192.168.1.56	192.168.1.55	ICMP	98	Echo (ping) request id=0x65d0, seq=1/256, ttl=64 (rep
22	20.589838697	192.168.1.56	192.168.1.55	ICMP	98	Echo (ping) request id=0x4565, seq=1/256, ttl=64 (rep
36	25.615206045	192.168.1.56	192.168.1.55	ICMP	98	Echo (ping) request id=0x2c99, seq=1/256, ttl=64 (rep
42	30.637559362	192.168.1.56	192.168.1.55	ICMP	98	Echo (ping) request id=0xe1b2, seq=1/256, ttl=64 (rep
45	35.653425361	192.168.1.56	192.168.1.55	ICMP	98	Echo (ping) request id=0xd5af, seq=1/256, ttl=64 (rep
49	40.691636706	192.168.1.56	192.168.1.55	ICMP	98	Echo (ping) request id=0xf8de, seq=1/256, ttl=64 (rep
53	45.714781772	192.168.1.56	192.168.1.55	ICMP	98	Echo (ping) request id=0x18c9, seq=1/256, ttl=64 (rep
57	50.729402254	192.168.1.56	192.168.1.55	ICMP	98	Echo (ping) request id=0x8947, seq=1/256, ttl=64 (rep
62	55.863183977	192.168.1.56	192.168.1.55	ICMP	98	Echo (ping) request id=0x05cf, seq=1/256, ttl=64 (rep
67	60.912841870	192.168.1.56	192.168.1.55	ICMP	98	Echo (ping) request id=0x0a59, seq=1/256, ttl=64 (rep
71	65.931945244	192.168.1.56	192.168.1.55	ICMP	98	Echo (ping) request id=0x9811, seq=1/256, ttl=64 (rep
73	70.943928140	192.168.1.56	192.168.1.55	ICMP	98	Echo (ping) request id=0x1772, seq=1/256, ttl=64 (rep
76	76.191852064	192.168.1.56	192.168.1.55	ICMP	98	Echo (ping) request id=0x0a2, seq=1/256, ttl=64 (rep

קודם כל התחלנו לשלוח קצת פינגים ממכונת המוניטור בשעה 10:20 ב10.4.25



אז התחלנו את המתקפה ב10.4.25 בשעה 17:22

```
ron@ron-VirtualBox:~$ sudo python3 attack5.py
Starting attack...
```

והמתקפה נגמרה ב23:22 6 שעות

```
Sent 790000 packets
Sent 800000 packets
Sent 810000 packets
Sent 820000 packets
Sent 830000 packets
Sent 840000 packets
Sent 850000 packets
Sent 860000 packets
Sent 870000 packets
Sent 880000 packets
Sent 890000 packets
Sent 900000 packets
Sent 910000 packets
Sent 920000 packets
Sent 930000 packets
Sent 940000 packets
Sent 950000 packets
Sent 960000 packets
Sent 970000 packets
Sent 980000 packets
Sent 990000 packets
Sent 1000000 packets

Attack completed successfully!
Total time: 21461.79 seconds
Average packet send time: 21.27 ms
Standard Deviation: 27.81 ms
ron@ron-VirtualBox:~$
```

ונוצרו לנו קבצים הבאים

מהמוניטור : pings\_results\_p.txt בפורמט הבא

ובסופו כך

```
4204 Ping 4250: 1.21 ms
4205 Ping 4251: 0.73 ms
4206 Ping 4252: 0.76 ms
4207 Ping 4253: 1.89 ms
4208 Ping 4254: 1.59 ms
4209 Ping 4255: 1.08 ms
4210 Ping 4256: 1.50 ms
4211 Ping 4257: 0.90 ms
4212 Ping 4258: 0.83 ms
4213
4214 Average RTT: 1.58 ms
4215 Standard Deviation: 15.50 ms
```

```
Ping 1: 1.53 ms
Ping 2: 0.82 ms
Ping 3: 1.12 ms
Ping 4: 1.41 ms
Ping 5: 1.94 ms
Ping 6: 7.60 ms
Ping 7: 1.43 ms
Ping 8: 2.41 ms
Ping 9: 1.77 ms
Ping 10: 2.51 ms
Ping 11: 6.10 ms
Ping 12: 1.19 ms
Ping 13: 1.81 ms
Ping 14: 1.68 ms
Ping 15: 0.86 ms
Ping 16: 0.58 ms
```

מהתוקף : syns\_results\_y.txt בפורמט הבא :

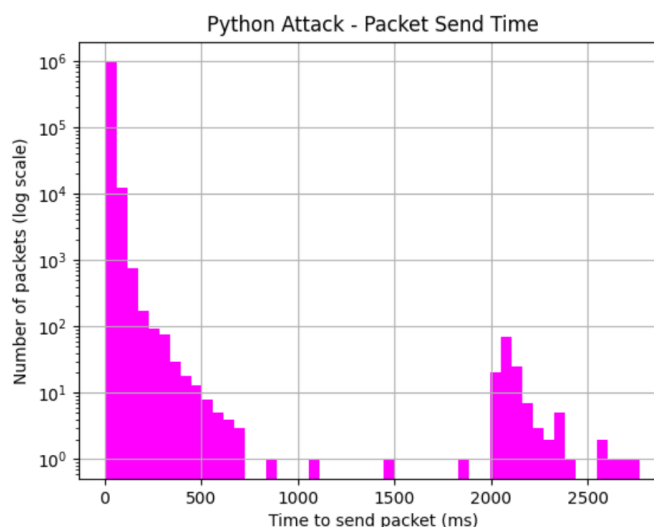
```
Packet 999998: 23.243 ms
Packet 999999: 19.546 ms
Packet 1000000: 23.069 ms

Total time: 24910.18 seconds
Average packet send time: 24.587 ms
Standard Deviation: 27.630 ms
```

```
Packet 1: 95.724 ms
Packet 2: 35.965 ms
Packet 3: 29.921 ms
Packet 4: 28.024 ms
Packet 5: 21.890 ms
Packet 6: 23.803 ms
Packet 7: 24.096 ms
Packet 8: 32.305 ms
Packet 9: 35.687 ms
Packet 10: 33.245 ms
Packet 11: 32.421 ms
Packet 12: 19.638 ms
Packet 13: 37.049 ms
Packet 14: 22.022 ms
Packet 15: 33.478 ms
Packet 16: 37.713 ms
```

גרפים:

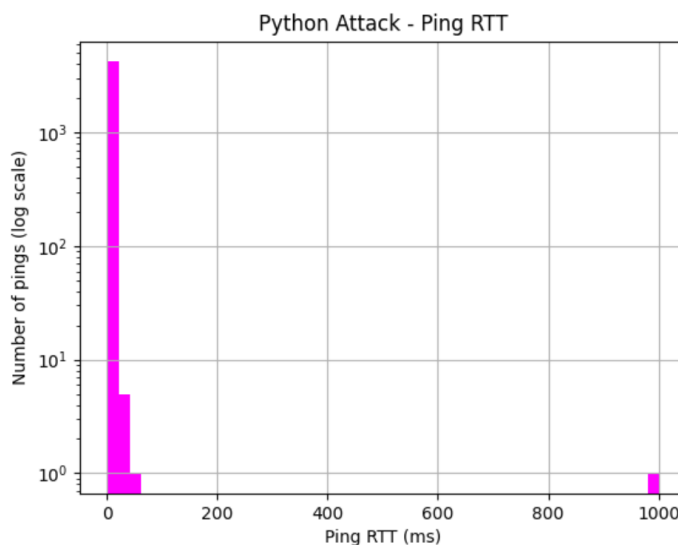
גרף הראשון : קצב שליחת SYN מצד התוקף



הגרף מציג את זמני השליחה של חבילות SYN במהלך המתקפה שבוצעה בפיתון. כל עמודה מייצגת את מספר החבילות שנשלחו בטווח זמן מסוים. מרבית הבקשות נשלחו בתוך פחות מ-100 מילישניות – דבר המעיד על מערכת תקינה בתחילת המתקפה. עם הזמן, ניתן לראות האטה הדרגתית, ולאחר מכן הופעה של קבוצת חבילות בודדות שנשלחו באיחור קיצוני של למעלה מ-2 שניות. תופעה זו מצביעה על עומס שהולך ונבנה במערכת התוקף.

ההתפלגות ממש רחבה ולא אחידה. רוב החבילות נשלחו יחסית מהר, אבל ככל שהזמן עבר, לקח יותר ויותר זמן לשלוח חבילה. אפילו יש חבילות שלקח להן יותר מ-2.5 שניות להישלח זה מראה שהמערכת התחילה טוב, אבל לאט לאט נחשפה לעומס על השרת.

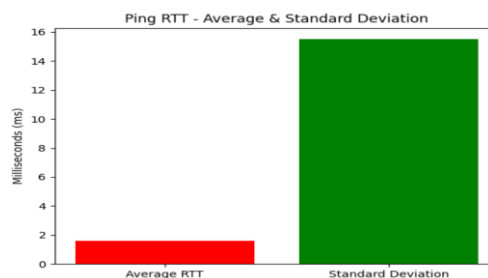
## גרף שני : המבטא את זמן תגובה של השרת (RTT)



הגרף מציג את זמני ה-RTT של פקודות ping לשרת במהלך המתקפה. בציר האופקי מופיע זמן התגובה של כל פינג (במילישניות), ובציר האנכי מספר הפינגים שנמדדו בזמן הזה. רוב הפינגים קיבלו תשובה מהירה, בזמן של פחות מ-20 ms אבל רואים גם פינגים בודדים שהגיעו לזמן תגובה של כמעט 1000ms המסקנה מהגרף היא שלמרות שהשרת המשיך להשיב לפינגים, המתקפה כן השפיעה על הביצועים שלו, כי ברגעים מסוימים נוצר עיכוב משמעותי בתגובה.

ככל שזמן שליחת החבילות עלה – בגלל עומס פנימי של הקוד או עומס שנוצר על השרת – כך גם נראו עיכובים בתשובות הפינג מהמוניטור. שני הגרפים יחד מראים שמערכת ההתקפה התחילה לעבוד מהר, אבל עם הזמן נכנסה לעומס כבד, מה שגרם גם להאטה בשליחה וגם לתגובה איטית מהשרת. זה מעיד על הידרדרות בביצועים לאורך זמן, שניתן לזהות גם מהשליחה וגם מהתגובה של השרת.

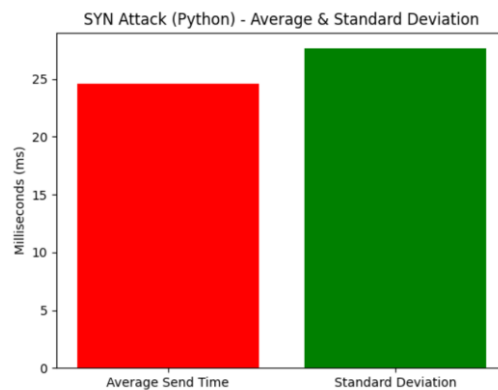
אנחנו יכולים להסתכל על הנתונים הסטטיסטיים שמופיעים בקבצי פלט שלנו להבין כי ניתוח סטטיסטי מהצד של התוקף : מדובר בזמן שליחה ממוצע נמוך יחסית, שמעיד על כך שהסקריפט מצליח לייצר תעבורה במהירות גבוהה. עם זאת, סטיית התקן הגבוהה מעידה על שונות גדולה בין זמני השליחה כלומר, למרות שרוב החבילות נשלחו מהר, חלקן נתקלו בהאטה. הדבר עשוי להעיד על עומס במערכת התוקפת או בממשק הרשת שלה.



ניתוח סטטיסטי מהצד של המוניטור : מדובר בממוצע RTT נמוך מאוד, אך סטיית התקן גדולה מאוד ביחס לממוצע. זה אומר שרוב הפינגים חזרו מהר מאוד, אבל היו מקרים בודדים שבהם ה-RTT עלה מאוד.

הממוצעים הנמוכים יכולים להטעות ולגרום לחשוב שלא הייתה השפעה, אבל דווקא סטיית התקן הגבוהה מצביעה על חוסר יציבות ברשת – זה בדיוק מה שמתקפת SYN Flood אמורה לגרום לו.

השרת לא קרס, אבל חווה ירידות רגעיות בביצועים שמתועדות בגרפים ובסטטיסטיקות.



הממוצעים הנמוכים עלולים להטעות וליצור תחושה שהשרת לא הושפע מההתקפה, אבל סטיות התקן הגבוהות מבהירות שכן הייתה **פגיעה ביציבות**. זו בדיוק המטרה של מתקפת – SYN Flood לאו דווקא להפיל את השרת, אלא לגרום לו להתאמץ, לאבד עקביות, ולהגיב לאט מדי פעם. השרת לא קרס, אבל בהחלט סבל מירידות רגעיות בביצועים וזה מתועד היטב בגרפים ובמדדים.

השוואה בין C לפיתון

כפי שידוע, שפת **C** היא שפת תכנות ברמת Low-Level קרובה מאוד לשפת מכונה, מה שמאפשר לה **לגשת ישירות לזיכרון**, לנהל משאבים בצורה מדויקת, ולבצע פעולות בקצב מהיר במיוחד. לעומתה, **פיתון** היא שפת High-Level שמתמקדת בנוחות קידוד אבל באה על חשבון **ביצועים וניהול יעיל של משאבים**.

מההשוואה שנעשתה, אפשר לראות בבירור שהתקפת SYN שנכתבה ב-C הייתה **יעילה משמעותית**:

- זמן השליחה הממוצע של חבילה היה נמוך ביותר כ-0.37ms והמערכת הצליחה לשלוח מיליון חבילות בתוך כמה דקות.
- זמני התגובה של השרת נשארו יציבים לאורך כל ההתקפה, ללא עיכובים מורגשים.

לעומת זאת, **פיתון**:

- דרשה מעל **24,000 שניות** (6 שעות) לשלוח את אותה כמות של חבילות.
  - הציגה **סטיות תקן גבוהות**, שמעידות על **חוסר יציבות בשליחה ובתגובה**.
  - יצרה עומס מורגש על השרת, שהתבטא בפיגים עם RTT חריגים – עד כדי ms1000
- פייתון נוחה כשכותבים קוד, אבל מהתוצאות שראינו היא לא מתאימה למטרות שדורשות מהירות כמו העמסת שרת בבקשות, בזמן ש-C הצליחה לשלוח מיליון חבילות בפחות מ-400 שניות, לפיתון לקח הרבה יותר זמן עם המון עיכובים לאורך הדרך. גם הפיגים מהמוניטור חזקו את זה ב-C הם נשארו יציבים, ובפייתון ראינו קפיצות לא צפויות.