

מטלה 2- רשתות תקשורת

1.

הסברים לקוד

Server

```
94 with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as server_socket:
95     # SO_REUSEADDR is a socket option that allows the socket to be bound to an address that is already in use.
96     server_socket.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
97
98     # Prepare the server socket
99     # * Fill in start (1)
100     server_socket.bind((host, port))
101     """
102     explanation-
103     connecting the socket to the host ip and port.
104     bind method prepares the server socket to listen for connection
105     on the specific port and address (ip address) and allow clients to connect to the socket.
106     """
```

פקודת BIND מחברת בין הSOCKET ש"נפתח" ובין הפורט והIP. מכינה את הSOCKET להאזנה בכתובת זו, ולהתחברות לCLIENTS.

```
107 server_socket.listen(1)
108     """
109     explanation-
110     listen method tells the server to wait for incoming connections.
111     the numeric param tells the socket the max number of client connections
112     he'll handle simultaneously. in this exercise the server will be handling
113     one client at max, so we set the param to 1.
114     """
115     """
```

פקודת LISTEN אומרת לשרת להאזין לבקשות התחברות נכנסות. הפרמטר המספרי מציין את מספר החיבורים שהשרת מאפשר לנהל במקביל. בחרנו ב-1 כיוון שבמטלה זו יתחבר רק לקוח אחד לשרת בכל זמן נתון.

```
116 server_socket.settimeout(1) # setting a timeout for to accept method. if quit was received,
117 # timeout will make sure new thread will not open
118 # * Fill in end (1)
```

היה לנו "באג" בסגירת הTHREADS כיוון שהשרת המשיך להאזין ולחכות לACCEPT. לכן הגדרנו TIMEOUT שיפסיק לחכות לACCEPT כאשר התקבלה בקשה לסגירת הSERVER (פונקציה שהוספנו בשביל נוחות בהרצת הקוד, שולחת הודעה עם בקשה לסגור את השרת וTERMINATE את התוכנה, על מנת שלא נצטרך לסגור בכל פעם ידנית את שלושת הקבצים).

```
157 # added lines-for quit option
158 except socket.timeout:
159     # accept() timed out, check if we should stop
160     if flag_quit:
161         # QUIT was signaled by a handler thread
162         break
163     # If not quitting, just continue waiting for connections.
164     pass
165 # end of added lines
```

בהקשר לפונקציית TIMEOUT, אם התקבלה הודעת סגירה- לא נמשיך לחכות לACCEPT, ונצא מהלולאה. במידה ולא התקבלה הודעה כזו- נישאר בלולאה (הודעת הסגירה היא עבור סגירת כל השרתים, ולא עבור התנתקות של לקוח ספציפי מהשרת. כמו שצוין- משמשת עבור נוחות שלנו בלבד ולא נדרשה במטלה).

```
169 # added lines-for terminating the program
170 try:
171     print("closing socket...")
172     server_socket.close() # Close the proxy socket
173     print("terminating server.py...")
174     sys.exit(0)
175 except Exception as e:
176     print(f"Error while closing server socket: {e}")
177 # end of added lines
178
```

סגירת התוכנה כאשר התקבלה הודעת QUIT, וכל ה-THREADS נסגרו. מוודא סגירה של ה-SOCKET למרות ש-WITH אמור לדאוג לכך לבד, כדי למנוע תקלות בסגירת הקובץ ולוודא שה-SOCKET באמת סגור.

```
90 # * Fill in start (3)
91 data = client_socket.recv(BUFSIZE)
92 """
93 explanation -
94 after a connection is established (accept method returned the client socket and address successfully)
95 a new thread is open in order to handle the communication with this specific client.
96 the recv method is waiting to receive data from the client.
97 reads up to BUFSIZE bytes of data from the client socket, if the client closed the connection- the recv
98 method will return an empty bytes object.
99 """
100
```

```
15 BUFSIZE = api.BUFFER_SIZE # using the API buffer size to ensure consistency in data handling across all socket operations
16
```

הפונקציה RECV משמשת לקבלת DATA מהלקוח. הפרמטר המספרי שהפונקציה מקבלת מסמן את הגודל המקסימלי של מידע שהפונקציה תאפשר לקבל בכל פעם. הגדרנו אותו כמשתנה BUFSIZE בראש התוכנית על מנת לאפשר את שינויו בקלות.

את המשתנה BUFSIZE הגדרנו לפי גודל הבאפר שהוגדר ב-API, על מנת לשמור על אחידות משתנה זה בכל הקבצים הנדרשים למטלה זו. בנוסף, זה מאפשר את שינויו בקלות לכל קבצי server.py, proxy.py, client.py במידה ויהיה בכך צורך.

```
201 try: # checking if QUIT message was received.
202     if len(data) == 4: # implemented in this scope since request is never shorter than 12 bytes,
203         # a 4 bytes data cant be a request.
204         is_quiting = data.decode("utf-8") # decoding bytes to string
205         print(is_quiting)
206         if "QUIT" in is_quiting:
207             flag_quit = True
208             print(flag_quit)
209             client_socket.close() # making sure client socket is closed
210             break
211 except Exception as e:
212     print(e)
213 # * Fill in end (3)
```

בלוק זה משמש עבור פונקציית הסגירה אותה ציינו קודם לכן. גודל הודעת הסגירה הוא 4, והוא בהכרח לא שווה לגודל בקשה אחרת שתתקבל (כיוון שזה קטן מהגודל המינימלי של פקטת הבקשה). כיוון שמבנה ההודעה שונה ממבנה של בקשה, השתמשנו בפונקציית

DECODE על מנת להמיר את ההודעה מ-BYTES ל-STRING. לאחר קבלת הודעה זו שינינו את המשתנה הגלובלי `flag_quit` ל-`TRUE`, על מנת לציין ללולאה הראשית שהתקבלה בקשה TO TERMINATE. סגירת ה-SOCKET נועדה על מנת לוודא שאכן נסגר, אל אף ש-WITH מטפלת בכך אוטומטית. לאחר מכן, התבצע BREAK על מנת לצאת מהלולאה ולהתחיל לטפל בסגירת הרצת הקובץ.

```
213     if not data: # * Change in start (1)
214         # exit loop when receiving no data (means that client closed the connection).
215         # no need to close client socket since the with method handles this automatically
216         break
217     # * Change in end (1)
```

במידה ולא התקבל מידע, זה סימן שהלקוח ניתק את הקשר עם השרת. לכן, נבצע BREAK מלולאת ה-`RECV`. המשך טיפול בסגירת הקשר- בהמשך הקוד.

```
233
234     # * Fill in start (4)
235     client_socket.sendall(response)
236     """
237     explanation -
238     there's two method we can choose from in order to send data to the client while using
239     TCP protocol:
240     send()- sends data up to some buffer size. it may not send all the data in one
241     call, so it will require us to handle the sending of the remaining data.
242     sendall- ensure the sending of the all the data, by internally loops, and handling the
243     buffer limitation automatically.
244     we chose to use sendall method in order to make sure all the data is send to the client.
245     """
246     # * Fill in end (4)
```

במידה ולא התקבלה בקשת QUIT, והתקבל מידע, המשכנו לפונקציית `process_request`. פונקציה זו החזירה לנו את התשובה שנשלח בחזרה ללקוח, ואותה הגדרנו במשתנה `response`. את התשובה נשלח ללקוח בעזרת פונקציית `sendall`. בחרנו בפונקציה זו כיוון שהיא מטפלת בשליחת תשובה בכל גודל בצורה אוטומטית, ולא נדרש להגדיר לה BUFFER ידנית.

```
252     # * Change in start (2)
253     print(f"{client_prefix} Connection closed")
254     client_socket.close() # same as line 254
255     return
256     # * Change in end (2)
257
```

בלוק זה נמצא מחוץ ללולאת ה-`RECV` ומחוץ לבלוק ה-WITH. כיוון שכאשר יצאנו מהלולאה הזו סימן שהלקוח התנתק, נדפיס הודעת התנתקות. הפקודה לסגירת ה-SOCKET אינה נחוצה, כיוון ש-WITH מטפל בכך אוטומטית, אך הוספנו ליתר בטחון (בעקבות באג בסגירת ה-THREADS, רצינו לוודא שאכן אין תקלה בסגירת ה-SOCKET שמשאירה את ה-THREAD פתוח).

Proxy

```
91 # Prepare the proxy socket
92 # * Fill in start (1)
93 proxy_socket.bind(proxy_address)
94 """
95 | see explanation about the bind method via server.py, line 97
96 | """
97 proxy_socket.listen()
98 """
99 | see explanation about the listen method via server.py, line 104
100 | """
101 # * Fill in end (1)
```

פקודות אלו הוסברו כבר בהסברי הקוד של הSERVER, וההסברים זהים לחלוטין גם כאן.

```
109 # * Fill in start (2)
110 proxy_socket.settimeout(1) # setting a timeout for to accept method. if quit was received,
111 # timeout will make sure new thread will not open
112 client_socket, client_address = proxy_socket.accept()
113 """
114 | see explanation about the accept method via server.py, line 120
115 | """
116 # * Fill in end (2)
```

גם פקודות אלו הוסברו כבר בהסברי הקוד של הSERVER, וההסברים זהים לחלוטין גם כאן. שינוי המיקום של תחילת הTIMEOUT אינו משנה לתפקוד פונקציה זו, והיא פועלת בדיוק כמו בSERVER על מנת להגדיר TIMEOUT לACCEPT.

```
126 # added lines-for quit option
127 except socket.timeout:
128     # accept() timed out, check if we should stop
129     if flag_quit:
130         # QUIT was signaled by a handler thread
131         break
132     # If not quitting, just continue waiting for connections.
133     pass
134 # end of added lines
```

גם שינוי זה זהה לחלוטין לSERVER.

```
139 if flag_quit: # after all threads where closed (finished handling the client) checking if a QUIT request was received
140     try:
141         with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as server_socket:
142             print(f"{proxy_address[0]}:{proxy_address[1]} requesting to terminate server.py")
143             server_socket.connect((server_host, server_port))
144             server_socket.sendall("QUIT".encode("utf-8"))
145     except Exception as e:
146         print(f"\nError while closing server: {e}")
147     print(f"{proxy_address[0]}:{proxy_address[1]} terminating proxy...")
148     try:
149         proxy_socket.close() # Close the proxy socket
150         sys.exit(0)
151     except Exception as e:
152         print(f"\nError while closing proxy socket: {e}")
153
```

טיפול בבקשת הסגירה וה TERMINATION של הרצת הקבצים. כאשר הPROXY מקבל בקשה לסגירה הוא מעביר את ההודעה לסגירה אל השרת, ולאחר מכן מטפל בסגירת הSOCKET שלו (של שרת הPROXY) ומסיים את הרצת התוכנית בעזרת פקודת `sys.exit(0)`. בוצע גם טיפול בשגיאות על ידי `try` ו-`except`, ובנוסף- הדפסת הודעות רלוונטיות בטרמינל בפורמט הנדרש.

```

165 # Receive data from the client
166 # * Fill in start (3) #
167 data = client_socket.recv(BUFSIZE)
168 """
169 see explanation about the recv method via server.py, line 120
170 here we used the same buffer size as used in the server code, since both the proxy socket and the server socket are
171 handling the same data size.
172 """

```

גם הסבר פונקציית RECV פה זהה להסבר הפונקציה המקבילה בSERVER, ולכן לא נסביר מחדש.

```

173 # checking if QUIT message was received.
174 if len(data) == 4: # implemented in this scope since request is never shorter than 12 bytes,
175     # a 4 bytes data cant be a request.
176     try:
177         is_quiting = data.decode("utf-8") # decoding bytes to string
178         if "QUIT" in is_quiting:
179             # when QUIT is received, break out of the loop to close the connection, send quit req to server
180             flag_quit = True
181             print(f"{client_prefix} Received QUIT, closing connection") # $ Added line $
182             break
183     except Exception as e:
184         print(e)
185 # * Fill in end (3)
186

```

טיפול בבקשת QUIT, גם כאן- ההסבר זהה להסבר הבולק המקביל בSERVER.

```

186
187 if not data: # * Change in start (1)
188     break
189 # * Change in end (1)

```

גם הסבר בלוק זה מקביל להסבר של בלוק השרת.

```

218
219 # Send the response back to the client
220 # * Fill in start (4)
221 client_socket.sendall(response)
222 """
223 see explanation about the accept method via server.py, line 199
224 """
225 # * Fill in end (4)
226

```

גם כאן, הסבר מקביל להסבר אצל השרת.

```

232 # * Change in start (2)
233 print(f"{client_prefix} Connection closed")
234 # * Change in end (2)
235

```

כאשר נצא מלולאת RECV, ולאחר מכן מבלוק הWITH נדפיס הודעת התנתקות מהלקוח. דילגנו על שאר הפקודות שהיו בקוד של הSERVER כיוון שהן אינן נחוצות, והוספנו אותן שם ליתר בטחון.

Client

```
63 def client(server_address: tuple[str, int], expressions_list: list[api.Expression], show_steps: bool = False,
64             cache_result: bool = False, cache_control: int = api.CalculatorHeader.MAX_CACHE_CONTROL) -> None:
```

שינוי מהקוד המקורי- הפונקציה מקבלת רשימת ביטויים ולא ביטוי בודד. כך ניתן לשלוח מספר ביטויים ולקבל עליהם תגובה מבלי להתחבר ולהתנתק משרת הPROXY לאחר כל שליחת בקשה וקבלת תגובה בודדת.

```
69         for expression in expressions_list:
70             try:
```

הכנסנו לולאת FOR בתוך בלוק WITH, כל מנת לשלוח את הביטויים אחד אחרי השני כאשר החיבור בין השרת והלקוח נשמר.

```
90 # Added lines: message to close proxy and server.
91 def closing_message(server_address: tuple[str, int]):
92     with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as client_socket:
93         client_socket.connect(server_address)
94         try:
95             message = "QUIT"
96             print("Sending request to close server")
97             client_socket.sendall(message.encode("utf-8"))
98         except Exception as exception:
99             print(exception)
100
101
102 # and of added lines
```

פונקציה שהוספנו על מנת לשלוח הודעת סגירה לפרוקסי (והפרוקסי מעביר את ההודעה הזו לשרת). הודעת סגירה זו אומרת לפרוקסי (ולאחר מכן לשרת) לסגור את SOCKETS ולאחר מכן TO TERMINATE את התוכנה. זה מאפשר סגירה נוחה יותר לאחר הרצה של שלושת הקבצים, וכך לא נדרש לסגור כל קובץ במקביל. לא נדרש לממש זאת במטלה.

```
148 # * Change in start (2)
149 """
150     List of expressions to choose from.
151 """
152 exprList = [
153     # mul_b(div_b(sin_f(max_f(2, mul_b(3, 4), 5, mul_b(div_b(mul_b(7, 8), 9)), div_b(10, 11))), 12), 13),
154     add_b(max_f(2, 3), 3),
155     add_b(3, div_b(mul_b(4, 2), pow_b(sub_b(1, 5), pow_b(2, 3)))),
156     div_b(pow_b(add_b(1, 2), mul_b(3, 4)), mul_b(5, 6)),
157     neg_u(neg_u(pow_b(add_b(1, add_b(2, 3)), neg_u(add_b(4, 5)))))
158     max_f(2, mul_b(3, 4), log_f(e_c), mul_b(6, 7), div_b(9, 8))
159 ]
160
161 """
162     making an expressions list to send to the server.
163     the quit option was implemented to make the termination of the program easier for the programmer,
164     and reducing the number of files that needs too be manually closed after each run.
165 """
```

נמצא בMAIN. הכנסנו את דוגמאות הביטויים לרשימה, וממנה הלקוח יכול לבחור ביטויים לשליחה. שינוי זה בוצע על מנת להקל על שליחת הביטויים.

```

167 exit = ""
168
169 while exit != "-1":
170     expToSend = []
171     print("Enter expression numbers from list to calculate, enter -1 to finish:")
172     expr = input("EXP: ")
173     while expr != "-1":
174         try:
175             expToSend.append(exprList[int(expr)])
176         except Exception as e:
177             print("illegal value")
178             print(e)
179         expr = input("EXP: ")
180
181     try:
182         client((api.DEFAULT_PROXY_HOST, api.DEFAULT_PROXY_PORT), expToSend, show_steps, cache_result,
183               cache_control)
184     except Exception as e:
185         print("illegal value")
186         print(e)
187
188     print("\nInstructions: "
189           "\nType -1 to close the connection, press ENTER to send a new request")
190     exit = input("")
191
192 terminate = input("Type -1 to terminate proxy.py, server.py, client.py: ")
193 if terminate == "-1":
194     closing_message((api.DEFAULT_PROXY_HOST, api.DEFAULT_PROXY_PORT))
195 # * Change in end (2)
196

```

לולאה שמטפלת בשליחת הביטויים. מהלך הלולאה:

1. לולאה עוטפת: כל עוד לא התקבלה בקשה לסיום:

א. לולאה פנימית: כל עוד לא התקבלה בקשה לשליחת הבקשות:

- (i) נקלוט את הביטוי מהלקוח (על ידי קליטת מיקומו ברשימה)
- (ii) נוסיף את הביטוי לרשימת הביטויים לשליחה
- (iii) אם התקבל -1, נצא מהלולאה.

לאחר קבלת בקשה לשליחת הבקשות (הוכנס -1):

- ב. נעביר את רשימת הביטויים לשליחה אל פונקציית הCLIENT
- ג. פונקציית הלקוח תטפל בשליחת הבקשות וקבלת התגובות.
- ד. בסיום הטיפול, נשאל את הלקוח האם ירצה לסיים את שליחת הבקשות, או האם ירצה לשלוח בקשה נוספת:

- אם מעוניין בשליחת בקשה נוספת- נישאר בלולאה (נחזור לסעיף א)
- אם מעוניין בסיום השליחה (הוכנס -1) - נצא מהלולאה

2. במידה והלקוח ביקש לסיים את שליחת הבקשות, נשאל האם ירצה לבצע

TERMINATION לכל קבצי ההרצה של התוכנית:

- אם לא – שרת הPROXY יישאר פתוח, הרצת קובץ הCLIENT תסתיים.
- אם כן (הוכנס 1) – נקרא לפונקציית closing_message שתשלח בקשת סגירה לPROXY (ולאחר מכן, PROXY ישלח בקשת סגירה גם לSERVER).

דוגמא להדפסות הטרימינל:

1. הכנסת שלושה ביטויים לשליחה:

```
Run: proxy x server x client x
C:\Users\thele\AppData\Local\Programs\Python\Python39\python.exe C:\Users\thele\PycharmProjects\Networks_EX2_final\client.py
Enter expression numbers from list to calculate, enter -1 to finish:
EXP: 1
EXP: 2
EXP: 3
```

2. הכנסת 1- לסיום קליטת הביטויים:

```
Run: proxy x server x client x
{127.0.0.1:9998} Sending request of length 548 bytes
{127.0.0.1:9998} Got response of length 222 bytes
Result: 3.0001220703125
Steps:
3 + ((4 * 2) / ((1 - 5) ** (2 ** 3))) = 3 + (8 / ((1 - 5) ** (2 ** 3)))
= 3 + (8 / (-4 ** (2 ** 3)))
= 3 + (8 / (-4 ** 8))
= 3 + (8 / 65536)
= 3 + 0.0001220703125
= 3.0001220703125
{127.0.0.1:9998} Connection closed

Instructions:
Type -1 to close the connection, press ENTER to send a new request
```

נשים לב- קיבלנו את התגובה לכל הבקשות שהוכנסו, תוך שמירה על הקשר עם השרת. ניתן לקשר התבצע רק לאחר שליחת כל הביטויים ברשימה וקבלת התגובה אליהם.

3. נדגים שליחה של 2 ביטויים חדשים:

```
Enter expression numbers from list to calculate, enter -1 to finish:
EXP: 1
EXP: 3
```

```
= 3 + 0.0001220703125
= 3.0001220703125
{127.0.0.1:9998} Sending request of length 506 bytes
{127.0.0.1:9998} Got response of length 236 bytes
Result: 9.92290301275212e-08
Steps:
-(((1 + (2 + 3)) ** -(4 + 5))) = -(((1 + 5) ** -(4 + 5)))
= -((6 ** -(4 + 5)))
= -((6 ** -(9)))
= -((6 ** -9))
= -((9.92290301275212e-08))
= -(-9.92290301275212e-08)
= 9.92290301275212e-08
{127.0.0.1:9998} Connection closed

Instructions:
Type -1 to close the connection, press ENTER to send a new request
```


4. כעת, נקליד 1- שוב על מנת לציין שאין לנו עוד בקשה לשלוח:

```
Run: proxy x server x client x
= -(-(6 ** -(9)))
= -(-(6 ** -9))
= -(-(9.92290301275212e-08))
= -(-(9.92290301275212e-08))
= 9.92290301275212e-08
{127.0.0.1:9998} Connection closed

Instructions:
Type -1 to close the connection, press ENTER to send a new request
-1
Type -1 to terminate proxy.py, server.py, client.py: |
```

נשים לב- בשלב זה הSERVER והPROXY עדיין פתוחים ומאפשרים קבלה של בקשות.

5. נקליד שוב 1- על מנת לבקש TERMINATION :

```
Run: proxy x server x client x
= 9.92290301275212e-08
{127.0.0.1:9998} Connection closed

Instructions:
Type -1 to close the connection, press ENTER to send a new request
-1
Type -1 to terminate proxy.py, server.py, client.py: -1
Sending request to close server

Process finished with exit code 0
|
```

נשים לב- כעת גם הSERVER וגם הPROXY סגורים, ובוצע TERMINATION להרצות שלהם כצפוי.

נציג גם את הדפסות הPROXY:

1. שלושת הבקשות הראשונות. ניתן לראות שהחיבור נשמר עבור שלושת הבקשות, והתנתק רק בסיום שליחת התגובה לבקשה האחרונה.

```
Run: proxy x server x client x
C:\Users\thele\AppData\Local\Programs\Python\Python39\python.exe C:\Users\thele\PycharmProjects\Network
Listening on 127.0.0.1:9998
{127.0.0.1:63371} Connected established
{127.0.0.1:63371} Got request of length 548 bytes
{127.0.0.1:63371} Cache miss, response cached ,server time remaining: inf, client time remaining: inf
{127.0.0.1:63371} Sending response of length 222 bytes
{127.0.0.1:63371} Got request of length 487 bytes
{127.0.0.1:63371} Cache miss, response cached ,server time remaining: inf, client time remaining: inf
{127.0.0.1:63371} Sending response of length 164 bytes
{127.0.0.1:63371} Got request of length 548 bytes
{127.0.0.1:63371} Cache hit ,server time remaining: inf, client time remaining: inf
{127.0.0.1:63371} Sending response of length 222 bytes
{127.0.0.1:63371} Connection closed
```

2. שתי הבקשות הנוספות:

```
{127.0.0.1:63371} Connection closed
{127.0.0.1:63628} Connected established
{127.0.0.1:63628} Got request of length 548 bytes
{127.0.0.1:63628} Cache hit ,server time remaining: inf, client time remaining: inf
{127.0.0.1:63628} Sending response of length 222 bytes
{127.0.0.1:63628} Got request of length 506 bytes
{127.0.0.1:63628} Cache miss, response cached ,server time remaining: inf, client time remaining: inf
{127.0.0.1:63628} Sending response of length 236 bytes
{127.0.0.1:63628} Connection closed
```

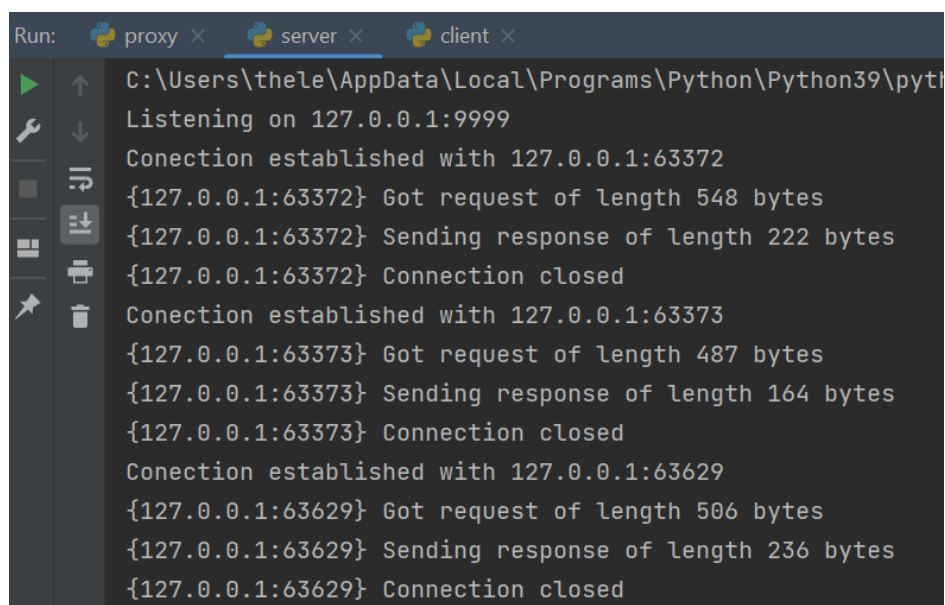
3. בקשת הTERMINATION. ניתן לראות שנשלחה בקשת סגירה לשרת ורק לאחר מכן סגרנו את הPROXY.

```
{127.0.0.1:63628} Connection closed
{127.0.0.1:64198} Connected established
{127.0.0.1:64198} Received QUIT, closing connection
{127.0.0.1:64198} Connection closed
127.0.0.1:9998 requesting to terminate server.py
127.0.0.1:9998 terminating proxy...

Process finished with exit code 0
```

נציג גם את הדפסות הSERVER:

1. חמשת הבקשות הראשונות-



```
Run: proxy x server x client x
C:\Users\thele\AppData\Local\Programs\Python\Python39\python.exe
Listening on 127.0.0.1:9999
Connection established with 127.0.0.1:63372
{127.0.0.1:63372} Got request of length 548 bytes
{127.0.0.1:63372} Sending response of length 222 bytes
{127.0.0.1:63372} Connection closed
Connection established with 127.0.0.1:63373
{127.0.0.1:63373} Got request of length 487 bytes
{127.0.0.1:63373} Sending response of length 164 bytes
{127.0.0.1:63373} Connection closed
Connection established with 127.0.0.1:63629
{127.0.0.1:63629} Got request of length 506 bytes
{127.0.0.1:63629} Sending response of length 236 bytes
{127.0.0.1:63629} Connection closed
```

נשים לב- אכן יצרנו קשר עם הSERVER רק כאשר נשלחו בקשות חדשות שהתגובה אליהן עדיין לא נשמרה בCACHE.

2. בקשת הTERMINATION :

```
{127.0.0.1:64201} Connection closed
```

```
Conection established with 127.0.0.1:64201
```

```
QUIT
```

```
{127.0.0.1:64201} Connection closed
```

```
closing socket...
```

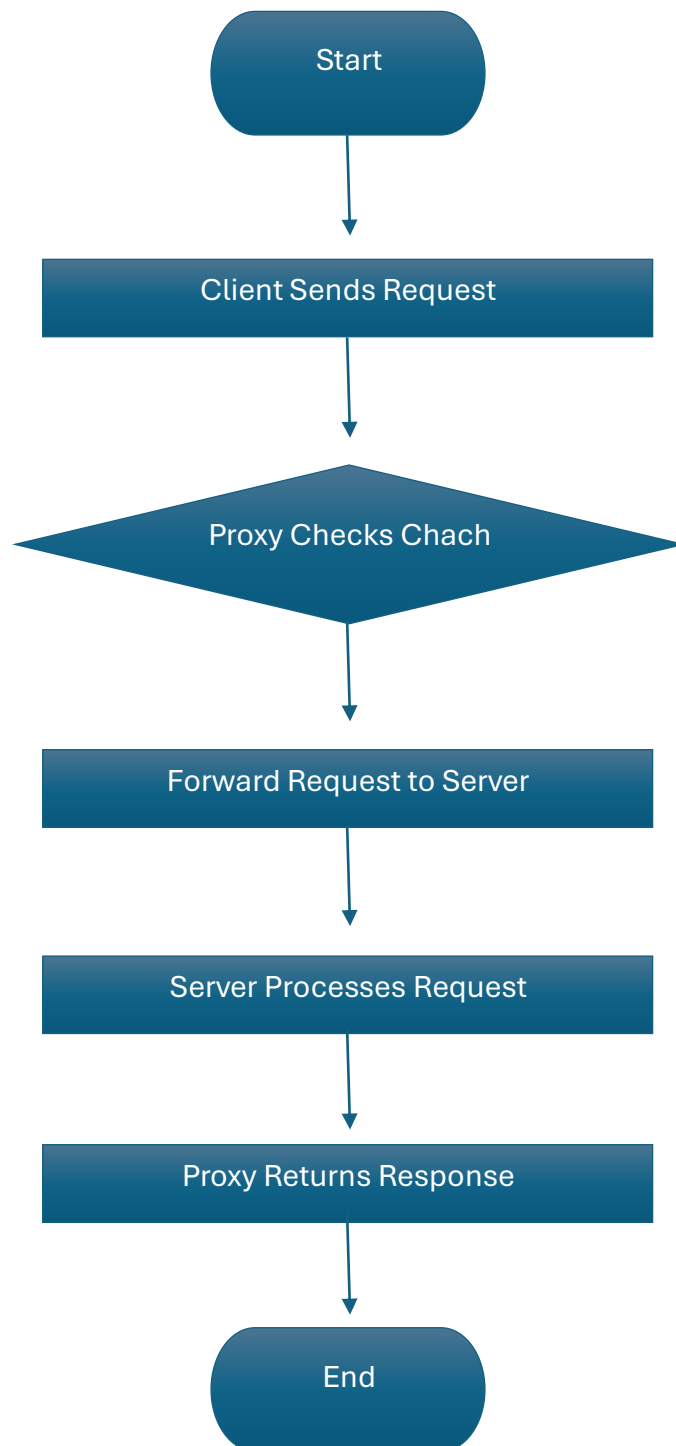
```
terminating server.py...
```

```
Process finished with exit code 0
```

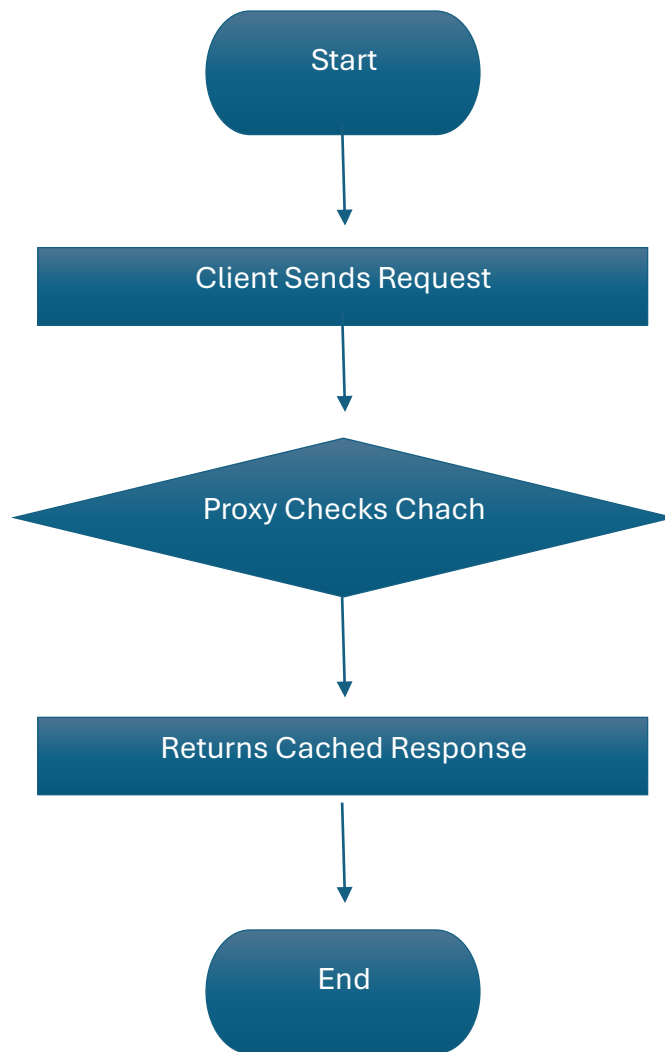
2.

תרשים זרימה להמחשת מערכת המטמון

2.1 האובייקט לא קיים במטמון



2.2 האובייקט קיים במטמון



הצגת הרצות ב- Wireshark

3.1 הרצה ראשונה-

הרצת הCLIENT והSERVER ושליחת אותה הבקשה פעמיים ואז בקשה שונה-

Client

```
Run: proxy x server x client x
{127.0.0.1:9999} Connection established
{127.0.0.1:9999} Sending request of length 548 bytes
{127.0.0.1:9999} Got response of length 222 bytes
Result: 3.0001220703125
Steps:

$$3 + ((4 * 2) / ((1 - 5) ** (2 ** 3))) = 3 + (8 / ((1 - 5) ** (2 ** 3)))$$


$$= 3 + (8 / (-4 ** (2 ** 3)))$$


$$= 3 + (8 / (-4 ** 8))$$


$$= 3 + (8 / 65536)$$


$$= 3 + 0.0001220703125$$


$$= 3.0001220703125$$

{127.0.0.1:9999} Sending request of length 548 bytes
{127.0.0.1:9999} Got response of length 222 bytes
Result: 3.0001220703125
Steps:

$$3 + ((4 * 2) / ((1 - 5) ** (2 ** 3))) = 3 + (8 / ((1 - 5) ** (2 ** 3)))$$


$$= 3 + (8 / (-4 ** (2 ** 3)))$$


$$= 3 + (8 / (-4 ** 8))$$


$$= 3 + (8 / 65536)$$


$$= 3 + 0.0001220703125$$


$$= 3.0001220703125$$

{127.0.0.1:9999} Sending request of length 487 bytes
{127.0.0.1:9999} Got response of length 164 bytes
Result: 17714.7
Steps:

$$((1 + 2) ** (3 * 4)) / (5 * 6) = (3 ** (3 * 4)) / (5 * 6)$$


$$= (3 ** 12) / (5 * 6)$$


$$= 531441 / (5 * 6)$$


$$= 531441 / 30$$


$$= 17714.7$$

{127.0.0.1:9999} Connection closed
```

Server

```
Run: proxy x server x client x
C:\Users\thele\AppData\Local\Programs\Python\Python39\python.exe C:\Users\thele\PycharmProjects\
Listening on 127.0.0.1:9999
Conection established with 127.0.0.1:60581
{127.0.0.1:60581} Got request of length 548 bytes
{127.0.0.1:60581} Sending response of length 222 bytes
{127.0.0.1:60581} Got request of length 548 bytes
{127.0.0.1:60581} Sending response of length 222 bytes
{127.0.0.1:60581} Got request of length 487 bytes
{127.0.0.1:60581} Sending response of length 164 bytes
{127.0.0.1:60581} Connection closed
```

נבחן את הקלטת ה-Wireshark - סינון לפי פורט 9999 (SERVER)

client_and_server.pcapng									
Help Tools Wireless Telephony Statistics Analyze Capture Go View Edit File									
tcp.port == 9999									
No.	Source	Protocol	Length	Info					
632	127.0.0.1	TCP	56	Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM [SYN] 9999 → 60581					
633	127.0.0.1	TCP	56	Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM [SYN, ACK] 60581 → 9999					
634	127.0.0.1	TCP	44	Seq=1 Ack=1 Win=2161152 Len=0 [ACK] 9999 → 60581					
635	127.0.0.1	TCP	592	Seq=1 Ack=1 Win=2161152 Len=548 [PSH, ACK] 9999 → 60581					
636	127.0.0.1	TCP	44	Seq=1 Ack=549 Win=2160640 Len=0 [ACK] 60581 → 9999					
637	127.0.0.1	TCP	266	Seq=1 Ack=549 Win=2160640 Len=222 [PSH, ACK] 60581 → 9999					
638	127.0.0.1	TCP	44	Seq=549 Ack=223 Win=2160896 Len=0 [ACK] 9999 → 60581					
639	127.0.0.1	TCP	592	Seq=549 Ack=223 Win=2160896 Len=548 [PSH, ACK] 9999 → 60581					
640	127.0.0.1	TCP	44	Seq=223 Ack=1097 Win=2160128 Len=0 [ACK] 60581 → 9999					
641	127.0.0.1	TCP	266	Seq=223 Ack=1097 Win=2160128 Len=222 [PSH, ACK] 60581 → 9999					
642	127.0.0.1	TCP	44	Seq=1097 Ack=445 Win=2160640 Len=0 [ACK] 9999 → 60581					
643	127.0.0.1	TCP	531	Seq=1097 Ack=445 Win=2160640 Len=487 [PSH, ACK] 9999 → 60581					
644	127.0.0.1	TCP	44	Seq=445 Ack=1584 Win=2159616 Len=0 [ACK] 60581 → 9999					
645	127.0.0.1	TCP	208	Seq=445 Ack=1584 Win=2159616 Len=164 [PSH, ACK] 60581 → 9999					
646	127.0.0.1	TCP	44	Seq=1584 Ack=609 Win=2160640 Len=0 [ACK] 9999 → 60581					
647	127.0.0.1	TCP	44	Seq=1584 Ack=609 Win=2160640 Len=0 [FIN, ACK] 9999 → 60581					
648	127.0.0.1	TCP	44	Seq=609 Ack=1585 Win=2159616 Len=0 [ACK] 60581 → 9999					
649	127.0.0.1	TCP	44	Seq=609 Ack=1585 Win=2159616 Len=0 [FIN, ACK] 60581 → 9999					
650	127.0.0.1	TCP	44	Seq=1585 Ack=610 Win=2160640 Len=0 [ACK] 9999 → 60581					

הסבר

יצירת החיבור בין השרת והלקוח:

Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM [SYN] 9999 → 60581	56	TCP	127.0.0.1	632
Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM [SYN, ACK] 60581 → 9999	56	TCP	127.0.0.1	633
Seq=1 Ack=1 Win=2161152 Len=0 [ACK] 9999 → 60581	44	TCP	127.0.0.1	634

פקטה ראשונה (632)- פקטת SYN שנשלחה מהלקוח והתקבלה אצל השרת, משמשת ליצירת הקשר הראשוני בין השרת והלקוח.

פקטה שניה (633)- פקטת SYN-ACK שנשלחה מהשרת והתקבלה אצל הלקוח. מאשרת את קבלת פקטת ה-SYN.

פקטה שלישית (634)- ACK מהלקוח לשרת, שמאשר את קבלת ה-SYN-ACK.

לאחר התהליך הזה מתבצעת לחיצת היד בין הלקוח והשרת.

בקשה ותגובה ראשונה -

בקשה:

Seq=1 Ack=1 Win=2161152 Len=548 [PSH, ACK] 9999 → 60581	592	TCP	127.0.0.1	635
Seq=1 Ack=549 Win=2160640 Len=0 [ACK] 60581 → 9999	44	TCP	127.0.0.1	636

(365) פקטת הבקשה שנשלחה מהלקוח אל השרת.

(336) ACK שנשלח מהשרת אל הלקוח ומאשרת את קבלת הבקשה.

Frame 635: 592 bytes on wire (4736 bits), 592 bytes captured (4736 bits) on interface \Device\NPF_{Loopback}, id 0 <									
Null/Loopback <									
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1 <									
Transmission Control Protocol, Src Port: 9999, Seq: 1, Ack: 1, Len: 548 <									
Data (548 bytes) <									

נשים לב- אורך הבקשה הראשונה לפי ההדפסות 548 bytes ואכן, אורך הדאטה בפקטה זו הוא 548 bytes כנדרש.

תגובה-

Seq=1 Ack=549 Win=2160640 Len=222 [PSH, ACK] 60581 → 9999	266	TCP	127.0.0.1 637
Seq=549 Ack=223 Win=2160896 Len=0 [ACK] 9999 → 60581	44	TCP	127.0.0.1 638

(367) פקטת התגובה שנשלחה מהשרת אל הלקוח.

(338) ACK שנשלח מהלקוח אל השרת ומאשרת את קבלת התגובה.

Frame 637: 266 bytes on wire (2128 bits), 266 bytes captured (2128 bits) on interface \Device\NPF_{Loopback, id 0	<
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1	<
Transmission Control Protocol, Src Port: 9999, Dst Port: 60581, Seq: 1, Ack: 549, Len: 222	<
Data (222 bytes)	<

נשים לב- אורך התגובה הראשונה לפי ההדפסות 222 bytes ואכן, אורך הדאטה בפקטה זו הוא 222 bytes כנדרש.

בקשה ותגובה שניה + שלישית -

המבנה זהה לבקשה הראשונה, לכן לא נסביר מחדש.

בקשה שניה:

Seq=549 Ack=223 Win=2160896 Len=548 [PSH, ACK] 9999 → 60581	592	TCP	127.0.0.1 639
Seq=223 Ack=1097 Win=2160128 Len=0 [ACK] 60581 → 9999	44	TCP	127.0.0.1 640
Seq=223 Ack=1097 Win=2160128 Len=222 [PSH, ACK] 60581 → 9999	266	TCP	127.0.0.1 641
Seq=1097 Ack=445 Win=2160640 Len=0 [ACK] 9999 → 60581	44	TCP	127.0.0.1 642

פקטת בקשה-

Frame 639: 592 bytes on wire (4736 bits), 592 bytes captured (4736 bits) on interface \Device\NPF_{Loopback, id 0	<
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1	<
Transmission Control Protocol, Src Port: 60581, Dst Port: 9999, Seq: 549, Ack: 223, Len: 548	<
Data (548 bytes)	<

פקטת תגובה-

A screenshot of the Wireshark network protocol analyzer. The top bar shows the file path: 'Wireshark · Packet 641 · client_and_server.pcapng'. The main pane displays the details of packet 641. The first line indicates the packet size: 'Frame 641: 266 bytes on wire (2128 bits), 266 bytes captured (2128 bits) on interface \Device\NPF_{Loopback, id 0'. The second line shows the network layer protocol: 'Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1'. The third line shows the transport layer protocol: 'Transmission Control Protocol, Src Port: 9999, Dst Port: 60581, Seq: 223, Ack: 1097, Len: 222'. The final line indicates the payload: 'Data (222 bytes)'. The interface name is partially visible as '\Device\NPF_{Loopback, id 0'.

נשים לב- אורך הבקשה השניה 548 bytes ואורך התגובה השלישית לפי ההדפסות 222 bytes ואכן, אורך הדאטה בפקטות האלו הוא 548 bytes ו- 222 bytes כנדרש.

בקשה שלישית:

Seq=1097 Ack=445 Win=2160640 Len=487 [PSH, ACK] 9999 → 60581	531	TCP	127.0.0.1 643
Seq=445 Ack=1584 Win=2159616 Len=0 [ACK] 60581 → 9999	44	TCP	127.0.0.1 644
Seq=445 Ack=1584 Win=2159616 Len=164 [PSH, ACK] 60581 → 9999	208	TCP	127.0.0.1 645
Seq=1584 Ack=609 Win=2160640 Len=0 [ACK] 9999 → 60581	44	TCP	127.0.0.1 646

פקטת בקשה-


```

Frame 643: 531 bytes on wire (4248 bits), 531 bytes captured (4248 bits) on interface \Device\NPF_{Loopback, id 0 <
Null/Loopback <
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1 <
Transmission Control Protocol, Src Port: 60581, Dst Port: 9999, Seq: 1097, Ack: 445, Len: 487 <
Data (487 bytes) <

```

פקטת תגובה-

```

Frame 645: 208 bytes on wire (1664 bits), 208 bytes captured (1664 bits) on interface \Device\NPF_{Loopback, id 0 <
Null/Loopback <
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1 <
Transmission Control Protocol, Src Port: 9999, Dst Port: 60581, Seq: 445, Ack: 1584, Len: 164 <
Data (164 bytes) <

```

נשים לב- אורך הבקשה השלישית 487 bytes ואורך התגובה השלישית לפי ההדפסות 164 bytes ואכן, אורך הדאטה בפקטות האלו הוא 487 bytes ו- 164 bytes כנדרש.

סיום הקשר בין השרת והלקוח-

Seq=1584 Ack=609 Win=2160640 Len=0 [FIN, ACK] 9999 → 60581 44	TCP	127.0.0.1 647
Seq=609 Ack=1585 Win=2159616 Len=0 [ACK] 60581 → 9999 44	TCP	127.0.0.1 648
Seq=609 Ack=1585 Win=2159616 Len=0 [FIN, ACK] 60581 → 9999 44	TCP	127.0.0.1 649
Seq=1585 Ack=610 Win=2160640 Len=0 [ACK] 9999 → 60581 44	TCP	127.0.0.1 650

(647) הלקוח שולח לשרת בקשה לסיים את התקשורת (פקטת FIN), כלומר- מעדכן שאין לו עוד דאטה לשלוח.

(648) ACK מהשרת ללקוח שמאשר את קבלת פקטת הFIN.

(649) השרת שולח ללקוח פקטת FIN על מנת לאשר שאין לו עוד דאטה לשלוח.

(650) הלקוח שולח לשרת ACK שמאשר את קבלת פקטת הFIN.

לאחר התהליך הזה הלקוח מתנתק מהשרת. נשים לב- השרת אינו "נסגר" ועדיין מאשר התחברות מחדש של הלקוח או התחברות של לקוחות אחרים.

3.2 הרצה שניה-

הרצת CLIENT, PROX ו-SERVER ושליחת אותה הבקשה פעמיים ואז בקשה שונה-

Client

```
Run: proxy x server x client x
{127.0.0.1:9998} Connection established
{127.0.0.1:9998} Sending request of length 548 bytes
{127.0.0.1:9998} Got response of length 222 bytes
Result: 3.0001220703125
Steps:
3 + ((4 * 2) / ((1 - 5) ** (2 ** 3))) = 3 + (8 / ((1 - 5) ** (2 ** 3)))
                                         = 3 + (8 / (-4 ** (2 ** 3)))
                                         = 3 + (8 / (-4 ** 8))
                                         = 3 + (8 / 65536)
                                         = 3 + 0.0001220703125
                                         = 3.0001220703125

{127.0.0.1:9998} Sending request of length 548 bytes
{127.0.0.1:9998} Got response of length 222 bytes
Result: 3.0001220703125
Steps:
3 + ((4 * 2) / ((1 - 5) ** (2 ** 3))) = 3 + (8 / ((1 - 5) ** (2 ** 3)))
                                         = 3 + (8 / (-4 ** (2 ** 3)))
                                         = 3 + (8 / (-4 ** 8))
                                         = 3 + (8 / 65536)
                                         = 3 + 0.0001220703125
                                         = 3.0001220703125

{127.0.0.1:9998} Sending request of length 487 bytes
{127.0.0.1:9998} Got response of length 164 bytes
Result: 17714.7
Steps:
((1 + 2) ** (3 * 4)) / (5 * 6) = (3 ** (3 * 4)) / (5 * 6)
                                = (3 ** 12) / (5 * 6)
                                = 531441 / (5 * 6)
                                = 531441 / 30
                                = 17714.7

{127.0.0.1:9998} Connection closed
```

Server

```
Run: proxy x server x client x
C:\Users\thele\AppData\Local\Programs\Python\Python39\python.exe
Listening on 127.0.0.1:9999
Conenction established with 127.0.0.1:63217
{127.0.0.1:63217} Got request of length 548 bytes
{127.0.0.1:63217} Sending response of length 222 bytes
{127.0.0.1:63217} Connection closed
Conenction established with 127.0.0.1:63218
{127.0.0.1:63218} Got request of length 487 bytes
{127.0.0.1:63218} Sending response of length 164 bytes
{127.0.0.1:63218} Connection closed
```

Proxy

```

Run: proxy x server x client x
C:\Users\thele\AppData\Local\Programs\Python\Python39\python.exe C:\Users\thele\PycharmProjects\Networks_EX
Listening on 127.0.0.1:9998
{127.0.0.1:63216} Connected established
{127.0.0.1:63216} Got request of length 548 bytes
{127.0.0.1:63216} Cache miss, response cached ,server time remaining: inf, client time remaining: inf
{127.0.0.1:63216} Sending response of length 222 bytes
{127.0.0.1:63216} Got request of length 548 bytes
{127.0.0.1:63216} Cache hit ,server time remaining: inf, client time remaining: inf
{127.0.0.1:63216} Sending response of length 222 bytes
{127.0.0.1:63216} Got request of length 487 bytes
{127.0.0.1:63216} Cache miss, response cached ,server time remaining: inf, client time remaining: inf
{127.0.0.1:63216} Sending response of length 164 bytes
{127.0.0.1:63216} Connection closed

```

נבחן את הקלטת ה Wireshark - סינון לפי פורט 9999 (SERVER) ולפי פורט 9998 - (PROXY)

יצירת החיבור בין השרת PROXY והלקוח:

tcp.port == 9999 tcp.port == 9998				
	Info	Length	Protocol	Source
	Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM [SYN] 9998 → 63216	56	TCP	127.0.0.1 221
	Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM [SYN, ACK] 63216 → 9998	56	TCP	127.0.0.1 222
	Seq=1 Ack=1 Win=2161152 Len=0 [ACK] 9998 → 63216	44	TCP	127.0.0.1 223

זוהי להרצה קודמת.

בקשה ותגובה ראשונה -

Seq=1 Ack=1 Win=2161152 Len=548 [PSH, ACK] 9998 → 63216	592	TCP	127.0.0.1 224
Seq=1 Ack=549 Win=2160640 Len=0 [ACK] 63216 → 9998	44	TCP	127.0.0.1 225
Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM [SYN] 9999 → 63217	56	TCP	127.0.0.1 226
Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM [SYN, ACK] 63217 → 9999	56	TCP	127.0.0.1 227
Seq=1 Ack=1 Win=2161152 Len=0 [ACK] 9999 → 63217	44	TCP	127.0.0.1 228
Seq=1 Ack=1 Win=2161152 Len=548 [PSH, ACK] 9999 → 63217	592	TCP	127.0.0.1 229
Seq=1 Ack=549 Win=2160640 Len=0 [ACK] 63217 → 9999	44	TCP	127.0.0.1 230
Seq=1 Ack=549 Win=2160640 Len=222 [PSH, ACK] 63217 → 9999	266	TCP	127.0.0.1 231
Seq=549 Ack=223 Win=2160896 Len=0 [ACK] 9999 → 63217	44	TCP	127.0.0.1 232
Seq=549 Ack=223 Win=2160896 Len=0 [FIN, ACK] 9999 → 63217	44	TCP	127.0.0.1 233
Seq=223 Ack=550 Win=2160640 Len=0 [ACK] 63217 → 9999	44	TCP	127.0.0.1 234
Seq=223 Ack=550 Win=2160640 Len=0 [FIN, ACK] 63217 → 9999	44	TCP	127.0.0.1 235
Seq=550 Ack=224 Win=2160896 Len=0 [ACK] 9999 → 63217	44	TCP	127.0.0.1 236
Seq=1 Ack=549 Win=2160640 Len=222 [PSH, ACK] 63216 → 9998	266	TCP	127.0.0.1 237
Seq=549 Ack=223 Win=2160896 Len=0 [ACK] 9998 → 63216	44	TCP	127.0.0.1 238

חלק ראשון- שליחת הבקשה לשרת PROXY-

Seq=1 Ack=1 Win=2161152 Len=548 [PSH, ACK] 9998 → 63216	592	TCP	127.0.0.1 224
Seq=1 Ack=549 Win=2160640 Len=0 [ACK] 63216 → 9998	44	TCP	127.0.0.1 225

פקטת הבקשה שנשלחה לפרוקסי-

Frame 224: 592 bytes on wire (4736 bits), 592 bytes captured (4736 bits) on interface \Device\NPF_{...}	
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1	
Transmission Control Protocol, Src Port: 63216, Dst Port: 9998, Seq: 1, Ack: 1, Len: 548	
Data (548 bytes)	

חלק שני- כיוון שהבקשה לא נמצאת במטמון, היא מועברת לSERVER.

א- יצירת החיבור עם השרת-

Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM [SYN] 9999 → 63217	56	TCP	127.0.0.1 226
Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM [SYN, ACK] 63217 → 9999	56	TCP	127.0.0.1 227
Seq=1 Ack=1 Win=2161152 Len=0 [ACK] 9999 → 63217	44	TCP	127.0.0.1 228

ב- העברת הבקשה לשרת-

Seq=1 Ack=1 Win=2161152 Len=548 [PSH, ACK] 9999 → 63217	592	TCP	127.0.0.1	229
Seq=1 Ack=549 Win=2160640 Len=0 [ACK] 63217 → 9999	44	TCP	127.0.0.1	230

Wireshark · Packet 229 · client_server_and_proxy.pcapng

Frame 229: 592 bytes on wire (4736 bits), 592 bytes captured (4736 bits) on interface \Device\NPF_{Loopback, id 0 <
 Null/Loopback <
 Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1 <
 Transmission Control Protocol, Src Port: 63217, Dst Port: 9999, Seq: 1, Ack: 1, Len: 548 <
 Data (548 bytes) <

ג- קבלת תגובה מהשרת-

Seq=1 Ack=549 Win=2160640 Len=222 [PSH, ACK] 63217 → 9999	266	TCP	127.0.0.1	231
Seq=549 Ack=223 Win=2160896 Len=0 [ACK] 9999 → 63217	44	TCP	127.0.0.1	232

פקטת התגובה

Wireshark · Packet 231 · client_server_and_proxy.pcapng

Frame 231: 266 bytes on wire (2128 bits), 266 bytes captured (2128 bits) on interface \Device\NPF_{Loopback, id 0 <
 Null/Loopback <
 Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1 <
 Transmission Control Protocol, Src Port: 9999, Dst Port: 63217, Seq: 1, Ack: 549, Len: 222 <
 Data (222 bytes) <

ד- סיום התקשורת עם השרת-

Seq=549 Ack=223 Win=2160896 Len=0 [FIN, ACK] 9999 → 63217	44	TCP	127.0.0.1	233
Seq=223 Ack=550 Win=2160640 Len=0 [ACK] 63217 → 9999	44	TCP	127.0.0.1	234
Seq=223 Ack=550 Win=2160640 Len=0 [FIN, ACK] 63217 → 9999	44	TCP	127.0.0.1	235
Seq=550 Ack=224 Win=2160896 Len=0 [ACK] 9999 → 63217	44	TCP	127.0.0.1	236

ה- העברת התגובה אל הלקוח-

Seq=1 Ack=549 Win=2160640 Len=222 [PSH, ACK] 63216 → 9998	266	TCP	127.0.0.1	237
Seq=549 Ack=223 Win=2160896 Len=0 [ACK] 9998 → 63216	44	TCP	127.0.0.1	238

פקטת התגובה

Wireshark · Packet 237 · client_server_and_proxy.pcapng

Frame 237: 266 bytes on wire (2128 bits), 266 bytes captured (2128 bits) on interface \Device\NPF_{Loopback, id 0 <
 Null/Loopback <
 Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1 <
 Transmission Control Protocol, Src Port: 9998, Dst Port: 63216, Seq: 1, Ack: 549, Len: 222 <
 Data (222 bytes) <

בקשה ותגובה שניה-

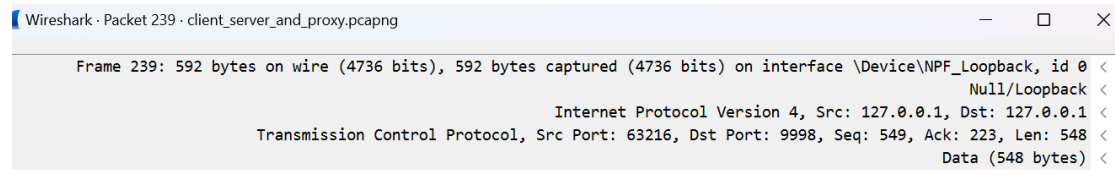
נצפה כמובן, שהתגובה תהיה במטמון ולכן שרת הפרוקסי יחזיר את התשובה ללא פניה לשרת.

Seq=549 Ack=223 Win=2160896 Len=548 [PSH, ACK] 9998 → 63216	592	TCP	127.0.0.1	239
Seq=223 Ack=1097 Win=2160128 Len=0 [ACK] 63216 → 9998	44	TCP	127.0.0.1	240
Seq=223 Ack=1097 Win=2160128 Len=222 [PSH, ACK] 63216 → 9998	266	TCP	127.0.0.1	241
Seq=1097 Ack=445 Win=2160640 Len=0 [ACK] 9998 → 63216	44	TCP	127.0.0.1	242

שליחת הבקשה לשרת PROXY-

Seq=549 Ack=223 Win=2160896 Len=548 [PSH, ACK] 9998 → 63216	592	TCP	127.0.0.1	239
Seq=223 Ack=1097 Win=2160128 Len=0 [ACK] 63216 → 9998	44	TCP	127.0.0.1	240

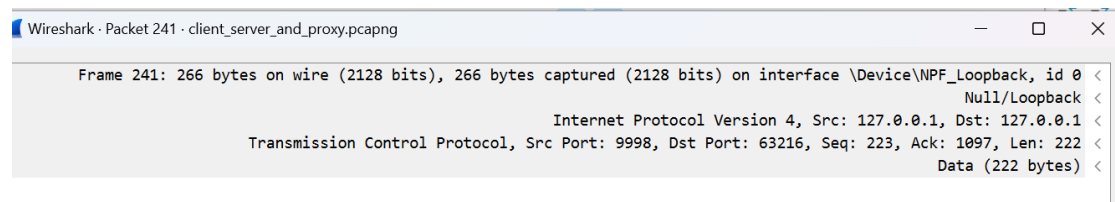
פקטת הבקשה



קבלת תגובה משרת ה-PROXY

Seq=223 Ack=1097 Win=2160128 Len=222 [PSH, ACK] 63216 → 9998	266	TCP	127.0.0.1	241
Seq=1097 Ack=445 Win=2160640 Len=0 [ACK] 9998 → 63216	44	TCP	127.0.0.1	242

פקטת התגובה



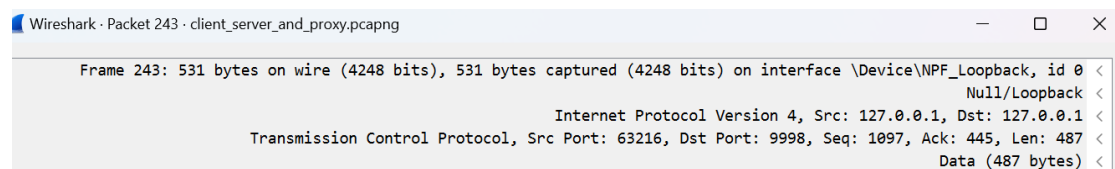
בקשה ותגובה שלישית

Seq=1097 Ack=445 Win=2160640 Len=487 [PSH, ACK] 9998 → 63216	531	TCP	127.0.0.1	243
Seq=445 Ack=1584 Win=2159616 Len=0 [ACK] 63216 → 9998	44	TCP	127.0.0.1	244
Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM [SYN] 9999 → 63218	56	TCP	127.0.0.1	245
Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM [SYN, ACK] 63218 → 9999	56	TCP	127.0.0.1	246
Seq=1 Ack=1 Win=2161152 Len=0 [ACK] 9999 → 63218	44	TCP	127.0.0.1	247
Seq=1 Ack=1 Win=2161152 Len=487 [PSH, ACK] 9999 → 63218	531	TCP	127.0.0.1	248
Seq=1 Ack=488 Win=2160640 Len=0 [ACK] 63218 → 9999	44	TCP	127.0.0.1	249
Seq=1 Ack=488 Win=2160640 Len=164 [PSH, ACK] 63218 → 9999	208	TCP	127.0.0.1	250
Seq=488 Ack=165 Win=2161152 Len=0 [ACK] 9999 → 63218	44	TCP	127.0.0.1	251
Seq=488 Ack=165 Win=2161152 Len=0 [FIN, ACK] 9999 → 63218	44	TCP	127.0.0.1	252
Seq=165 Ack=489 Win=2160640 Len=0 [ACK] 63218 → 9999	44	TCP	127.0.0.1	253
Seq=165 Ack=489 Win=2160640 Len=0 [FIN, ACK] 63218 → 9999	44	TCP	127.0.0.1	254
Seq=489 Ack=166 Win=2161152 Len=0 [ACK] 9999 → 63218	44	TCP	127.0.0.1	255

חלק ראשון - שליחת הבקשה לשרת ה-PROXY

Seq=1097 Ack=445 Win=2160640 Len=487 [PSH, ACK] 9998 → 63216	531	TCP	127.0.0.1	243
Seq=445 Ack=1584 Win=2159616 Len=0 [ACK] 63216 → 9998	44	TCP	127.0.0.1	244

פקטת הבקשה שנשלחה לפרוקסי



חלק שני - כיוון שהבקשה לא נמצאת במטמון, היא מועברת לSERVER.

א- יצירת החיבור עם השרת-

Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM [SYN] 9999 → 63218	56	TCP	127.0.0.1	245
Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM [SYN, ACK] 63218 → 9999	56	TCP	127.0.0.1	246
Seq=1 Ack=1 Win=2161152 Len=0 [ACK] 9999 → 63218	44	TCP	127.0.0.1	247

ב- העברת הבקשה לשרת-

Seq=1 Ack=1 Win=2161152 Len=487 [PSH, ACK] 9999 → 63218	531	TCP	127.0.0.1	248
Seq=1 Ack=488 Win=2160640 Len=0 [ACK] 63218 → 9999	44	TCP	127.0.0.1	249

פקטת הבקשה שנשלחה לשרת

Wireshark · Packet 248 · client_server_and_proxy.pcapng

Frame 248: 531 bytes on wire (4248 bits), 531 bytes captured (4248 bits) on interface \Device\NPF_{Loopback}, id 0 < Null/Loopback < Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1 < Transmission Control Protocol, Src Port: 63218, Dst Port: 9999, Seq: 1, Ack: 1, Len: 487 < Data (487 bytes) <

ג- קבלת תגובה מהשרת-

Seq=1 Ack=488 Win=2160640 Len=164 [PSH, ACK] 63218 → 9999	208	TCP	127.0.0.1	250
Seq=488 Ack=165 Win=2161152 Len=0 [ACK] 9999 → 63218	44	TCP	127.0.0.1	251

פקטת התגובה

Wireshark · Packet 250 · client_server_and_proxy.pcapng

Frame 250: 208 bytes on wire (1664 bits), 208 bytes captured (1664 bits) on interface \Device\NPF_{Loopback}, id 0 < Null/Loopback < Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1 < Transmission Control Protocol, Src Port: 9999, Dst Port: 63218, Seq: 1, Ack: 488, Len: 164 < Data (164 bytes) <

ד- סיום התקשורת עם השרת-

Seq=488 Ack=165 Win=2161152 Len=0 [FIN, ACK] 9999 → 63218	44	TCP	127.0.0.1	252
Seq=165 Ack=489 Win=2160640 Len=0 [ACK] 63218 → 9999	44	TCP	127.0.0.1	253
Seq=165 Ack=489 Win=2160640 Len=0 [FIN, ACK] 63218 → 9999	44	TCP	127.0.0.1	254
Seq=489 Ack=166 Win=2161152 Len=0 [ACK] 9999 → 63218	44	TCP	127.0.0.1	255

ה- העברת התגובה אל הלקוח-

Seq=445 Ack=1584 Win=2159616 Len=164 [PSH, ACK] 63216 → 9998	208	TCP	127.0.0.1	256
Seq=1584 Ack=609 Win=2160640 Len=0 [ACK] 9998 → 63216	44	TCP	127.0.0.1	257

פקטת התגובה

Wireshark · Packet 256 · client_server_and_proxy.pcapng

Frame 256: 208 bytes on wire (1664 bits), 208 bytes captured (1664 bits) on interface \Device\NPF_{Loopback}, id 0 < Null/Loopback < Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1 < Transmission Control Protocol, Src Port: 9998, Dst Port: 63216, Seq: 445, Ack: 1584, Len: 164 < Data (164 bytes) <

סיום הקשר בין שרת הפרוקסי ללקוח-

Seq=1584 Ack=609 Win=2160640 Len=0 [FIN, ACK] 9998 → 63216	44	TCP	127.0.0.1	258
Seq=609 Ack=1585 Win=2159616 Len=0 [ACK] 63216 → 9998	44	TCP	127.0.0.1	259
Seq=609 Ack=1585 Win=2159616 Len=0 [FIN, ACK] 63216 → 9998	44	TCP	127.0.0.1	260
Seq=1585 Ack=610 Win=2160640 Len=0 [ACK] 9998 → 63216	44	TCP	127.0.0.1	261

3.3 הרצה שלישית-

הרצת CLIENT, PROX ו-SERVER ושליחת בקשה, סגירת ה-SERVER, שליחת הבקשה שוב ואז שליחת בקשה שונה.

Client

```
Run: proxy x client x
{127.0.0.1:9998} Connection established
{127.0.0.1:9998} Sending request of length 548 bytes
{127.0.0.1:9998} Got response of length 222 bytes
Result: 3.0001220703125
Steps:
3 + ((4 * 2) / ((1 - 5) ** (2 ** 3))) = 3 + (8 / ((1 - 5) ** (2 ** 3)))
= 3 + (8 / (-4 ** (2 ** 3)))
= 3 + (8 / (-4 ** 8))
= 3 + (8 / 65536)
= 3 + 0.0001220703125
= 3.0001220703125
```

```
Run: proxy x client x
Enter expression numbers from list to calculate, enter -1 to finish:
EXP: 1
EXP: 2
EXP: 3
{127.0.0.1:9998} Connection established
{127.0.0.1:9998} Sending request of length 548 bytes
{127.0.0.1:9998} Got response of length 222 bytes
Result: 3.0001220703125
Steps:
3 + ((4 * 2) / ((1 - 5) ** (2 ** 3))) = 3 + (8 / ((1 - 5) ** (2 ** 3)))
= 3 + (8 / (-4 ** (2 ** 3)))
= 3 + (8 / (-4 ** 8))
= 3 + (8 / 65536)
= 3 + 0.0001220703125
= 3.0001220703125
{127.0.0.1:9998} Sending request of length 487 bytes
{127.0.0.1:9998} Got response of length 170 bytes
{127.0.0.1:9998} Got error: ('Internal proxy error', CalculatorServerError('Connection refused by server and the request was not in the cache/it was stale'))
{127.0.0.1:9998} Connection closed
```

Proxy

```
Run: proxy x client x
C:\Users\thele\AppData\Local\Programs\Python\Python39\python.exe C:\Users\thele\PycharmProjects\Networks_EX2_final\proxy.py
Listening on 127.0.0.1:9998
{127.0.0.1:65173} Connected established
{127.0.0.1:65173} Got request of length 548 bytes
{127.0.0.1:65173} Cache miss, response cached ,server time remaining: inf, client time remaining: inf
{127.0.0.1:65173} Sending response of length 222 bytes
{127.0.0.1:65173} Connection closed
{127.0.0.1:65185} Connected established
{127.0.0.1:65185} Got request of length 548 bytes
{127.0.0.1:65185} Cache hit ,server time remaining: inf, client time remaining: inf
{127.0.0.1:65185} Sending response of length 222 bytes
{127.0.0.1:65185} Got request of length 487 bytes
Unexpected server error: Connection refused by server and the request was not in the cache/it was stale
{127.0.0.1:65185} Connection closed
```

נבחן את הקלטת ה Wireshark- סינון לפי פורט 9999 (SERVER) ולפי פורט 9998
-(PROXY)

יצירת החיבור בין השרת PROXY והלקוח:

Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM [SYN] 9998 → 65173 56	TCP	127.0.0.1 218
Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM [SYN, ACK] 65173 → 9998 56	TCP	127.0.0.1 219
Seq=1 Ack=1 Win=2161152 Len=0 [ACK] 9998 → 65173 44	TCP	127.0.0.1 220

זוהי להרצה קודמת.

בקשה ותגובה ראשונה -

זוהי לשאלה קודמת.

Seq=1 Ack=1 Win=2161152 Len=548 [PSH, ACK]	9998 → 65173	592	TCP	127.0.0.1	221
Seq=1 Ack=549 Win=2160640 Len=0 [ACK]	65173 → 9998	44	TCP	127.0.0.1	222
Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM [SYN]	9999 → 65174	56	TCP	127.0.0.1	223
Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM [SYN, ACK]	65174 → 9999	56	TCP	127.0.0.1	224
Seq=1 Ack=1 Win=2161152 Len=0 [ACK]	9999 → 65174	44	TCP	127.0.0.1	225
Seq=1 Ack=1 Win=2161152 Len=548 [PSH, ACK]	9999 → 65174	592	TCP	127.0.0.1	226
Seq=1 Ack=549 Win=2160640 Len=0 [ACK]	65174 → 9999	44	TCP	127.0.0.1	227
Seq=1 Ack=549 Win=2160640 Len=222 [PSH, ACK]	65174 → 9999	266	TCP	127.0.0.1	228
Seq=549 Ack=223 Win=2160896 Len=0 [ACK]	9999 → 65174	44	TCP	127.0.0.1	229
Seq=549 Ack=223 Win=2160896 Len=0 [FIN, ACK]	9999 → 65174	44	TCP	127.0.0.1	230
Seq=223 Ack=550 Win=2160640 Len=0 [ACK]	65174 → 9999	44	TCP	127.0.0.1	231
Seq=223 Ack=550 Win=2160640 Len=0 [FIN, ACK]	65174 → 9999	44	TCP	127.0.0.1	232
Seq=550 Ack=224 Win=2160896 Len=0 [ACK]	9999 → 65174	44	TCP	127.0.0.1	233
Seq=1 Ack=549 Win=2160640 Len=222 [PSH, ACK]	65173 → 9998	266	TCP	127.0.0.1	234
Seq=549 Ack=223 Win=2160896 Len=0 [ACK]	9998 → 65173	44	TCP	127.0.0.1	235

221-222 – שליחת בקשה לפרוקסי

223-225 – חיבור לשרת

226-229 – העברת בקשה וקבלת תגובה מהשרת

230-233 – סיום התקשורת עם השרת

234-235 – העברת התגובה על ידי שרת הפרוקסי ללקוח

מקטע נוסף-

צורת השליחה של בקשות מהלקוח בקוד שלי מאמצת סגירה של החיבור עם שרת הPROXY והתחברות מחדש על מנת לסגור את הSERVER במהלך הרצה.

Seq=549 Ack=223 Win=2160896 Len=0 [FIN, ACK]	9998 → 65173	44	TCP	127.0.0.1	236
Seq=223 Ack=550 Win=2160640 Len=0 [ACK]	65173 → 9998	44	TCP	127.0.0.1	237
Seq=223 Ack=550 Win=2160640 Len=0 [FIN, ACK]	65173 → 9998	44	TCP	127.0.0.1	238
Seq=550 Ack=224 Win=2160896 Len=0 [ACK]	9998 → 65173	44	TCP	127.0.0.1	239
Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM [SYN]	9998 → 65185	56	TCP	127.0.0.1	507
Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM [SYN, ACK]	65185 → 9998	56	TCP	127.0.0.1	508
Seq=1 Ack=1 Win=2161152 Len=0 [ACK]	9998 → 65185	44	TCP	127.0.0.1	509

בקשה ותגובה שניה-

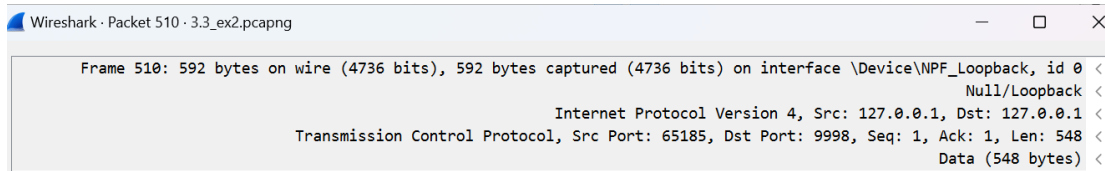
נצפה כמובן, שהתגובה תהיה במטמון ולכן שרת הפרוקסי יחזיר את התשובה ללא פניה לשרת. לכן סיום התקשורת עם השרת לא ישפיע על קבלת התגובה

Seq=1 Ack=1 Win=2161152 Len=548 [PSH, ACK]	9998 → 65185	592	TCP	127.0.0.1	510
Seq=1 Ack=549 Win=2160640 Len=0 [ACK]	65185 → 9998	44	TCP	127.0.0.1	511
Seq=1 Ack=549 Win=2160640 Len=222 [PSH, ACK]	65185 → 9998	266	TCP	127.0.0.1	512
Seq=549 Ack=223 Win=2160896 Len=0 [ACK]	9998 → 65185	44	TCP	127.0.0.1	513

שליחת הבקשה לשרת הPROXY-

510-511

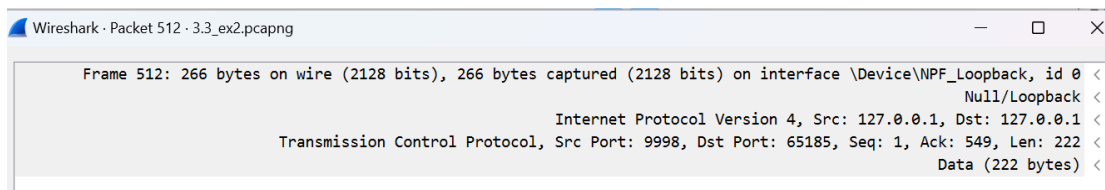
פקטת הבקשה



קבלת תגובה משרת הPROXY-

512-513

פקטת התגובה

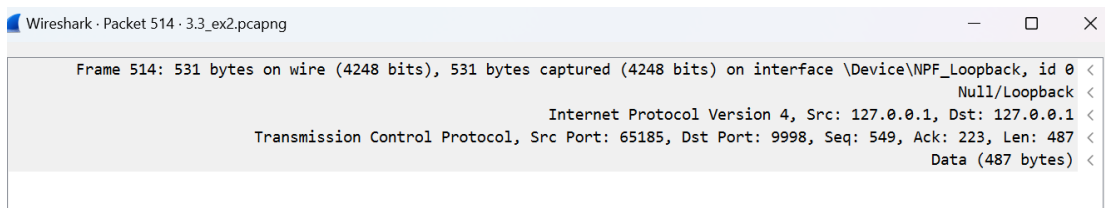


בקשה ותגובה שלישית-

חלק ראשון- שליחת הבקשה לשרת הPROXY-

Seq=549 Ack=223 Win=2160896 Len=487 [PSH, ACK]	9998 → 65185	531	TCP	127.0.0.1	514
Seq=223 Ack=1036 Win=2160128 Len=0 [ACK]	65185 → 9998	44	TCP	127.0.0.1	515

פקטת הבקשה שנשלחה לפרוקסי



חלק שני- כיוון שהבקשה לא נמצאת במטמון, היא מועברת לSERVER. כיוון שסגרנו את השרת נצפה לשגיאה.

א- בקשה ליצירת החיבור עם השרת-

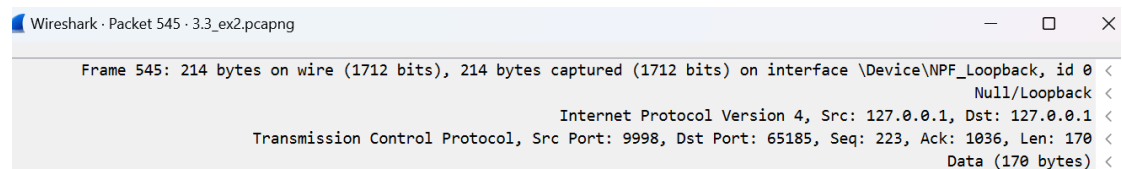
Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM [SYN]	9999 → 65186	56	TCP	127.0.0.1	516
--	--------------	----	-----	-----------	-----

ב- חוסר הצלחה להתחבר לשרת-

Seq=1 Ack=1 Win=0 Len=0 [RST, ACK]	65186 → 9999	44	TCP	127.0.0.1	517
Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM [TCP Port numbers reused]	56	TCP	127.0.0.1	522	
Seq=1 Ack=1 Win=0 Len=0 [RST, ACK]	65186 → 9999	44	TCP	127.0.0.1	523
Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM [TCP Port numbers reused]	56	TCP	127.0.0.1	524	
Seq=1 Ack=1 Win=0 Len=0 [RST, ACK]	65186 → 9999	44	TCP	127.0.0.1	525
Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM [TCP Port numbers reused]	56	TCP	127.0.0.1	541	
Seq=1 Ack=1 Win=0 Len=0 [RST, ACK]	65186 → 9999	44	TCP	127.0.0.1	542
Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM [TCP Port numbers reused]	56	TCP	127.0.0.1	543	
Seq=1 Ack=1 Win=0 Len=0 [RST, ACK]	65186 → 9999	44	TCP	127.0.0.1	544

ג- העברת השגיאה משרת הPROXY אל הלקוח-

Seq=223 Ack=1036 Win=2160128 Len=170 [PSH, ACK]	65185 → 9998	214	TCP	127.0.0.1	545
Seq=1036 Ack=393 Win=2160896 Len=0 [ACK]	9998 → 65185	44	TCP	127.0.0.1	546



סיום הקשר בין השרת ללקוח-

Seq=1036 Ack=393 Win=2160896 Len=0 [FIN, ACK]	9998 → 65185	44	TCP	127.0.0.1	547
Seq=393 Ack=1037 Win=2160128 Len=0 [ACK]	65185 → 9998	44	TCP	127.0.0.1	548
Seq=393 Ack=1037 Win=2160128 Len=0 [FIN, ACK]	65185 → 9998	44	TCP	127.0.0.1	549
Seq=1037 Ack=394 Win=2160896 Len=0 [ACK]	9998 → 65185	44	TCP	127.0.0.1	550

שאלות עיוניות

א. חישוב זמן טעינת הדף כולל התמונות

נתונים:

- גודל הדף: F Kbyte
- מספר תמונות: X
- גודל כל תמונה: 4F Kbyte
- זמן עיבוד של כל נתב: d
- קצב השידור: R
- אורך קו: x
- מהירות התפשטות בקווים: s

סה"כ גודל הדף והתמונות:

גודל הדף והתמונות = $F + X \cdot 4F = F \cdot (1 + 4X)$ נחשב קודם את RTT, T_{prop}, T_{tran}

$$T_{prop} = \text{Propagation Delay} = \frac{\text{length of the link}}{\text{velocity of a bit in the link}} =$$

$$\frac{\text{אורך קו}}{\text{מהירות ההתפשטות}} = \frac{x}{s}$$

$$T_{tran} = \text{Transmission Delay} = \frac{\text{Packet size}}{\text{Transmission rate}} =$$

$$\frac{\text{גודל תמונה}}{\text{קצב השידור}} = \frac{4F}{R}$$

$$RTT = d_{propagation} + d_{process} + d_{queue}$$

נחשב כל אחד מהגורמים קודם:

$$d_{propagation} = 2 \times \frac{\text{אורך קו}}{\text{מהירות ההתפשטות}} = \frac{2x}{s} \quad (\text{נכפול ב-2 בגלל ההלך חזור})$$

ב $d_{process}$ יש 2 נתבים, ובכל אחד זמן העיבוד הוא d .
מכיוון שהחבילה עוברת את הנתבים הלך וחזור נכפול שוב ב-2:

$$d_{process} = 2 \times 2 \times d = 4d$$

 d_{queue} לא נתון אז התייחסנו אליו כזניח:

$$d_{queue} = 0$$

לכן יוצא שRTT:

$$RTT = \frac{2x}{s} + 4d$$

עכשיו שיש את כל הנתונים אפשר לחשב עבור כל אחד מהמקרים:

1. HTTP Non-Persistent

במקרה הזה הזמן הוא:

$$\begin{aligned} RTT + RTT + T_{prop} + X(RTT + RTT + T_{tran}) = \\ \left(\frac{2x}{s} + 4d\right) + \left(\frac{2x}{s} + 4d\right) + \frac{x}{s} + X\left(\frac{2x}{s} + 4d + \frac{2x}{s} + 4d + \frac{4F}{R}\right) = \\ \left(\frac{4x}{s} + 8d\right) + \frac{x}{s} + X\left(\frac{4x}{s} + 8d + \frac{4F}{R}\right) = (X + 1)\left(\frac{4x}{s} + 8d\right) + \frac{x}{s} + \frac{4FX}{R} \end{aligned}$$

2. HTTP Persistent

כמו המקרה הראשון אך בלי תוספת RTT לכל אובייקט (כי דרך חיבור TCP אחד ניתן להעביר מספר אובייקטים)

ולכן הזמן הוא:

$$\begin{aligned} RTT + RTT + T_{prop} + X(RTT + T_{tran}) = \\ \left(\frac{2x}{s} + 4d\right) + \left(\frac{2x}{s} + 4d\right) + \frac{x}{s} + X\left(\frac{2x}{s} + 4d + \frac{4F}{R}\right) = \\ \left(\frac{4x}{s} + 8d\right) + \frac{x}{s} + X\left(\frac{2x}{s} + 4d + \frac{4F}{R}\right) \end{aligned}$$

3. HTTP Persistent with Pipelining

כמו המקרה הראשון אך בלי תוספת RTT לכל אובייקט (כי דרך חיבור TCP אחד ניתן להעביר מספר אובייקטים). כיוון שאנו עובדים עם שידור עקבי/צנרת, אז יש לנו אפשרות לשלוח את כל הבקשות אחת אחרי השנייה, דבר שיוצר שידור רצוף של כלל המידע מהשרת.

ולכן הזמן יהיה:

$$\begin{aligned} RTT + RTT + T_{prop} + RTT + X \cdot T_{tran} = 3RTT + T_{prop} + X \cdot T_{tran} = \\ 3 \cdot \left(\frac{2x}{s} + 4d\right) + \frac{x}{s} + \frac{4FX}{R} \end{aligned}$$

ב. הסבר ההבדלים בין השיטות

1. HTTP Non-Persistent:

- כל בקשה דורשת חיבור חדש, לכן זמן הטעינה עשוי להיות ארוך יותר.
- כל בקשה דורשת זמן של שידור, עיבוד ותקשורת מחדש עם השרת.

2. HTTP Persistent:

- אפשר לשלוח מספר בקשות על אותו חיבור.
- החיסכון הוא זמן החיבור והפחתת העומס של יצירת חיבורים חדשים.
- אבל עדיין יש עיכוב בין הבקשות (אף שהחיבורים קיימים).

3. HTTP Persistent with Pipelining:

- אפשר לשלוח מספר בקשות במקביל על אותו חיבור.
- יש שיפור משמעותי בזמן הטעינה כי יש פחות עיכוב בין הבקשות (לא צריך לחכות לסיום כל בקשה לפני שמבקשים את הבאה בתור).

ג. ההבדל בין הודעת HTTP POST להודעת HTTP GET:

• HTTP GET:

- התפקיד שלה לשלוח בקשה כדי לקרוא מידע מהשרת.
- הפרמטרים נשלחים כחלק מהכתובת, (URL) כלומר הם גלויים.
- משמשת בעיקר עבור פעולות קריאה שאינן משנות את המידע בשרת.

• HTTP POST:

- התפקיד שלה לשלוח נתונים לשרת, בדרך כלל כדי ליצור או לעדכן מידע.
- הנתונים נשלחים בתוך גוף הבקשה ולא ב URL-מה שמאפשר שליחה של נתונים גדולים ומורכבים יותר.
- משמשת לעדכון מצב או ביצוע פעולות שמשפיעות על המידע בשרת (למשל, שליחת טופס).

2. קודם נחשב כמה זמן יקח לשתף את כל החברים על ידי שימוש בשרת חזק:

כיוון שהשרת שולח לכל אחד ישירות, זמן ההעברה יהיה :

$$\text{Max} \left\{ (1) \text{Max} \frac{\text{לדוג צבוקה}}{\text{בצק הדרוהה לש לכ דחא מירבחה}}, (2) \frac{\text{לדוג צבוקה} * \text{רפסמ מירבחה}}{\text{בצק האלעהה לש תרשה}} \right\}$$

=

נחשב קודם את (1):

$$(1) \text{Max} \frac{\text{גודל הקובץ}}{\text{קצב ההורדה של כל אחד מהחברים}} = \frac{\text{גודל הקובץ}}{\text{הקטן מבין קצב ההורדה של החברים}}$$

הקצב הורדה הכי קטן הוא 4MB לשניה, גודל הקובץ הוא 1GB שזה 1000MB לכן:

$$\frac{1000}{4} = 250 \text{ שניות}$$

עכשיו נחשב את (2):

$$(2) \frac{\text{גודל הקובץ} * \text{מספר החברים}}{\text{קצב ההעלאה של השרת}} = \frac{30 * 1000}{100} = 300 \text{ שניות}$$

ניקח את המקסימלי מבין השניים שזה (2), לכן הזמן שייקח לשתף את כל החברים עם הקבצים זה 300 שניות.

עכשיו נחשב כמה זמן יקח לשתף את כל החברים על ידי שימוש ברשת P2P :

ברשת P2P, החברים לא רק מורידים את הקובץ מדן, אלא גם משתפים חלקים מהקובץ זה עם זה. זה אמור להגדיל את קצב ההפצה.

דן מתחיל להעלות חלקים מהקובץ לחברים במהירות שלו. ברגע שחבר קיבל חלק מהקובץ, הוא יכול לשתף את החלק הזה עם חברים אחרים. קצב ההפצה הכולל ברשת הוא סכום של קצבי ההעלאה של כל המשתתפים.

כדי למצוא את הזמן שייקח לדן לשתף את כל החברים בקובץ נצטרך לחשב:

$$\text{Max} \left\{ \begin{array}{l} (1) \text{Max} \frac{\text{גודל הקובץ}}{\text{קצב ההורדה של כל אחד מהחברים}}, (2) \frac{\text{גודל הקובץ}}{\text{קצב ההעלאה של דן}}, (3) \frac{\text{גודל הקובץ}}{\text{סכום קצבי ההעלאה של כל החברים}} \end{array} \right\}$$

את (1) חישבנו כבר קודם ויצא 250 שניות.

נחשב את (2), גודל הקובץ הוא 1GB = 1000MB, קצב ההעלאה של דן הוא 10MB לשניה, לכן:

$$(2) \frac{\text{גודל הקובץ}}{\text{קצב ההעלאה של דן}} = \frac{1000MB}{10MB \text{ לשניה}} = 100 \text{ שניות}$$

נחשב את (3), מכיון שיש 20 חברים עם קצב העלאה של 5MB לשניה ו10 חברים עם קצב העלאה של 2MB לשניה.

סה"כ ביחד זה יוצא שקצב ההעלאה של כל החברים יחד הוא 120MB לשניה. לכן:

$$(3) \frac{\text{גודל הקובץ}}{\text{סכום קצבי ההעלאה של כל החברים}} = \frac{1000MB}{120MB \text{ לשניה}} = 8.33 \text{ שניות}$$

אחרי שחישבנו את שלושתם יוצא:

$$\text{Max} \left\{ \text{Max} \frac{\text{גודל הקובץ}}{\text{קצב ההורדה של כל אחד מהחברים}}, \frac{\text{גודל הקובץ}}{\text{קצב ההעלאה של דן}}, \frac{\text{גודל הקובץ}}{\text{סכום קצבי ההעלאה של כל החברים}} \right\} = \text{Max}\{250, 100, 8.33\} = 250 \text{ שניות}$$

לכן, כדאי לדון להתחבר עם המחשב האישי שלו לרשת P2P כי הזמן שליחה של הקובץ לכל החברים יוצא יותר נמוך.

ב. 1. בשרת חזק הגורמים המשפיעים הוא קצב ההורדה של החבר עם הקצב הכי איטי או קצב ההעלאה של השרת.

במקרה שלנו, קצב ההעלאה של השרת מגביל את הזמן, אז נרצה לשפר את קצב ההעלאה של השרת (ואפשר בנוסף גם לשפר את קצב ההורדה של החברים אם רוצים לשפר עוד יותר).

במקרה הזה אם לדוגמה קצב ההעלאה של השרת יהיה 1GB לשניה:

$$30 = \frac{30 \times 1GB}{1GB \text{ לשניה}} \text{ שניות ואז המקסימלי מבין שני הדברים יהיה גורם מספר (1) שיצא 250.}$$

כמובן שגם אותו ניתן לשפר על ידי שיפור קצב ההורדה של כל החברים נניח ל10MB לשניה וכך יצא שזמן העברת הקובץ יהיה 1000/10 שזה 100 שניות.

1. ברשת P2P הגורמים המשפיעים הם:

קצבי ההעלאה של כל המשתתפים יחד- ככל שיש יותר משתתפים עם קצבי העלאה גבוהים, כך זמן ההפצה מתקצר.

קצב ההעלאה של שולח הקובץ – דן במקרה הזה.

קצב ההורדה של החבר עם קצב ההורדה הכי איטי.

כדי לשפר את הזמן הסופי אפשר להגדיל את קצב ההורדה של כל המשתתפים ל10MB לשניה, כך יצא שקצב הורדת הקובץ של החבר עם הקצב המינימלי יהיה גם ל10MB לשניה.

וכך:

$$Max \frac{\text{גודל הקובץ}}{\text{קצב ההורדה של כל אחד מהחברים}} = \frac{1000MB}{10MB \text{ לשניה}} = 100 \text{ שניות}$$

כך הקטנו את התוצאה הסופית שהיתה 250 שניות במקום להיות 100 שניות.

ג. הבעיות המרכזיות ברשתות P2P ופתרונות

1. בעיית עומס על משתמשים עם קצב העלאה נמוך

- **הבעיה:** ברשתות P2P, המשתתפים נדרשים להעלות חלקים מהקובץ לאחרים. אם להרבה משתתפים יש קצב העלאה נמוך, קצב ההפצה יהיה איטי יותר.
- **פתרון:** שימוש באלגוריתמים חכמים כמו BitTorrent, שנותנים עדיפות לשיתוף בין משתתפים עם קצבי העלאה גבוהים ומחלקים את הקובץ לחלקים קטנים מאוד, כך שכל משתתף יכול לשתף גם חלק קטן.

2. בעיית משתמשים חמדנים

- **הבעיה:** ישנם משתמשים שמורידים חלקים מהקובץ, אך לא משתפים חזרה. זה פוגע בקצב ההפצה הכללי.

- **פתרון:** משתמשים יוכלו להוריד חלקים רק אם הם ישתפו חלקים בעצמם. מנגנונים שגורמים לרצות לשתף (לדוגמה, לתת ציון גבוה למשתתפים שתורמים יותר).

4. בעיית אבטחה ופרטיות

- **הבעיה:** ברשת P2P קשה לשלוט על התכנים שמועברים, ויש סיכון לתוכן מזויף, וירוסים, או פגיעה בפרטיות.
- **הפתרון:** אימות קבצים באמצעות **Hashing** כדי לוודא שהתוכן שהורד תואם את המקום, שימוש בהצפנה כדי להגן על פרטיות המשתתפים.