

פרויקט סיום בתקשורת מחשבים

שני והב 208584557 נחשון בר סלע 318531290

על הפרויקט:

מטרת הפרויקט היא ליצור מעין צ'אט, אשר מספר לקוחות יכולים להתחבר לשרת בו זמנית ולשוחח ביניהם. בנוסף לכך, הם יכולים להתעדכן ברשימת המחוברים בזמן אמת לשרת, לקבצים הנמצאים בשרת ולהורדת הקבצים.

דרך העבודה:

תחילה יצרנו את השרת (*server.py*), אשר מחזיק את רשימת הלקוחות המחוברים אליו, הוא מעדכן את הרשימה כאשר לקוח מתחבר או מתנתק ומוציא הודעה מתאימה למשתמשים המחוברים.

השרת נענה לבקשות לקוח עפ"י ההודעה המגיעה אליו ממנו. לדוגמא, אם לקוח רוצה לדעת מי הם המשתמשים המחוברים כעת לשרת, הוא לוחץ על הכפתור הייעודי לכך "Online users" אשר מפעילה פונקציה השולחת הודעה מתאימה לשרת, ומכאן השרת מבין שעליו לשלוח ללקוח זה את רשימת המחוברים.

לאחר מכן, יצרנו את הלקוח (*client.py*), לכל לקוח יש מספר PORT בין 55000 ל-55015, ו-IP מקומי (localhost). לכל משתמש קיים צ'אט משלו וכל אחד מהלקוחות מסוגל לשלוח הודעה ל-server או להוריד קבצים ממנו, במקביל למשתמשים אחרים.

כל הודעה שמגיעה מהשרת ללקוח מוצגת בצ'אט המשתמש.

קבצים:

כחלק מהפרויקט, ביצענו הורדת קבצים מהשרת כאשר ההורדה עצמה מתבצעת באמצעות קשר UDP אמיני, לאחר כ-50% מהורדת הקובץ שאלנו את הלקוח האם הוא מעוניין להמשיך, במידה וכן, המשכנו את הורדת הקובץ ובסיומה הדפסנו למשתמש את ערך הבית האחרון.

תהליך הורדת הקובץ:

פתחנו מחלקה (*serverFile.py*) שתעבוד על העברת הקובץ מהשרת ע"י פרוטוקול UDP. לאחר מכן, חילקנו את כל הקובץ לסגמנטים בגודל 100 בית כל אחד, והכנסו לרשימה לפי הסדר המתאים. באמצע הרשימה "שתלנו" סגמנט פיקטיבי, על מנת שכאשר הלקוח יקבל אותו הוא יבין שברשותו כ-50% מהקובץ ולכן יוציא הודעה מתאימה למשתמש השואל אותו אם הוא רוצה להמשיך בהורדת הקובץ. במידה והמשתמש רוצה להמשיך ההורדה ממשיכה, ובסיומה מדפיסה את ערך הבית האחרון למשתמש.

אופן הורדת הקובץ בצורה אמינה:

על מנת שאופן העברת הקובץ מהשרת ללקוח תתנהל בצורה אמינה, כלומר שכל החבילות יעברו בסדר המתאים ושלא נאבד אף חבילה בדרך, לפני העברת החבילה השרת מחשב checksum של המידע ומעביר גם אותו באותה החבילה. כאשר החבילה מגיעה ללקוח, הלקוח גם הוא מחשב את checksum של המידע, במידה והתוצאה זהה לתוצאה שקיבל השרת ניתן להניח כי המידע הגיע ללא שגיאות.

על מנת לוודא שכל החבילות הגיעו וגם הגיעו בסדר הנכון, בנינו פרוטוקול Go-Back-N כאשר N (גודל החלון) הוא 4. השרת שולח ללקוח 4 חבילות אחת אחרי השנייה ומפעיל טיימר עם זמן מוקצב, כל חבילה מתויגת ע"י sequence number. הלקוח דורש לקבל את החבילות בסדר המתאים, ע"י שליחת הודעת 'ACK' מתאימה לשרת – אם הלקוח לדוגמא קיבל מהשרת חבילה מספר 3, הוא ישלח לו חזרה 'ACK 4' האומר "קיבלתי את חבילה מספר 3 ועכשיו אני רוצה לקבל אך ורק את חבילה מספר 4", כל חבילה אחרת שהיא לא חבילה מספר 4 הלקוח לא יסכים לקבל ופשוט יזרוק אותה וישלח מחדש 'ACK 4' לשרת. אופן העברת החבילות ממשיך כך עד שהלקוח

מבקש חבילה שלא קיימת, ואז השרת מעדכן אותו שכל המידע הועבר אליו.
במידה וחבילה נאבדה בדרך, כלומר השרת שלח אותה אבל היא לא הגיעה ללקוח, אם עובר מספיק זמן קורה timeout, כלומר הטיימר הסתיים ולכן השרת שולח מחדש את כל הסגמנטים שבחלון.

התמודדות עם עומסים ברשת:

על גבי פרוטוקול Go-Back-N בנינו Congestion control. באמצעות ניתוח הודעות ה-ACK' המגיעות מהלקוח אנחנו יכולים לדעת האם החבילות מגיעות אליו בסדר הנכון וללא שגיאות (במידה והוא תמיד מחזיר לנו 'ACK' הדורש את החבילה הבאה ולא אחת שכבר שלחנו). כל עוד החבילות מגיעות ללקוח בסדר הנכון וללא שגיאות, גודל החלון גדל ב-1.

אם התבצע timeout אנחנו מיידית מקטינים את גודל החלון ל-1, מכיוון ש- timeout מציג לנו מצב מאוד בעייתי ברשת בו חבילות הולכות לאיבוד או מתעכבות זמן רב. בין אם זה בגלל עומסים ברשת או בגלל כל גורם אחר. לעומת זאת, אם חבילות מגיעות ללקוח אבל עם שגיאות, אנחנו מקטינים את גודל החלון רק בחצי.

אופן פתיחת הצ'אט ושימוש בו:

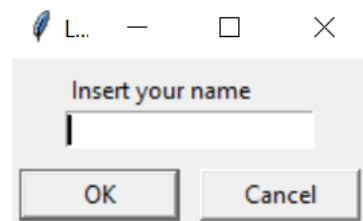
תחילה נריץ ב-Command Line את השרת ע"י הפקודה:

```
python server.py
```

ואת יצירת הצ'אט ע"י הפקודה:

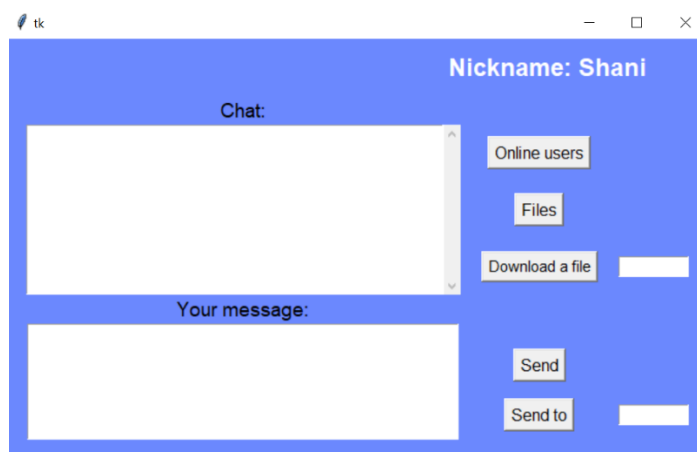
```
python newChat.py
```

לאחר הרצת פקודה זו, הלקוח מתחבר לשרת והשרת מבקש מהמשתמש להכניס nickname



זה השם עבורו הלקוח יוצג עבור שאר המשתמשים המחוברים.

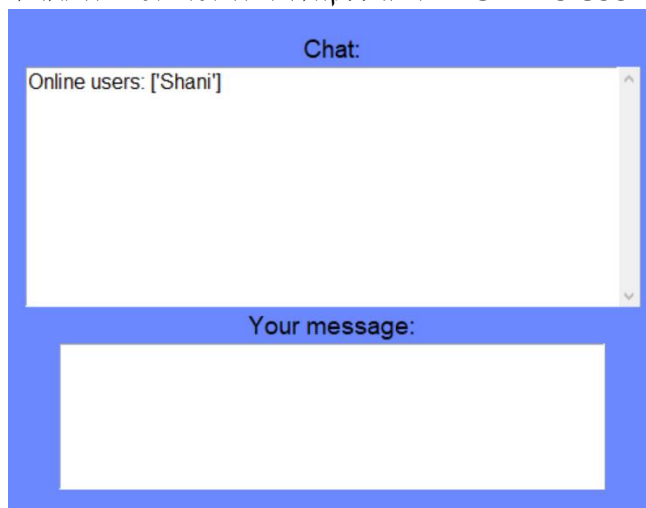
לאחר הכנסת הכינוי, הצ'אט המוצג למשתמש נראה כך :



כפתורים:



Online users – מציג ללקוח את המשתמשים המחוברים



Files – מציג ללקוח את הקבצים על השרת

Chat:

Online users: ['Shani']
Files: ['client.py', 'clientFile.py', 'file.txt', 'server.py', 'serverFile.py', 'todo.txt']

Your message:

Download a file – מאפשר הורדה של קובץ מתוך הקבצים שבשרת *יש לכתוב את שם הקובץ כולל הסימנת בתיבה המתאימה

Download a file

todo.txt

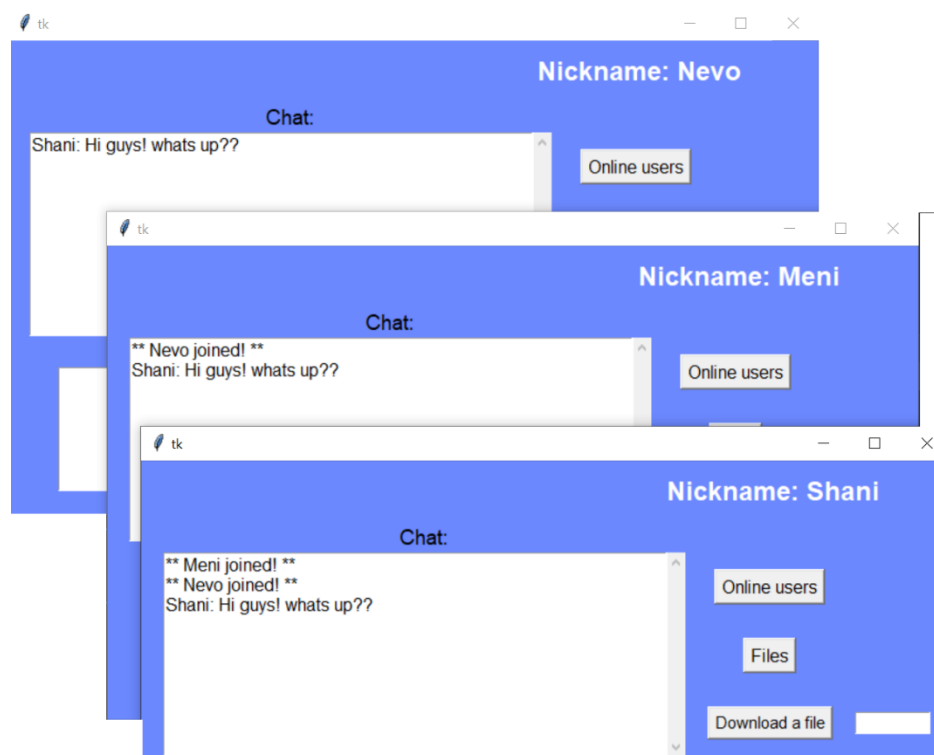
Chat:

Online users: ['Shani']
Files: ['client.py', 'clientFile.py', 'file.txt', 'server.py', 'serverFile.py', 'todo.txt']

File size = 59156
Downloading...
Continues to download...
The download is complete
Last byte in the file = Q

Your message:

Send – שליחת הודעה לכלל המשתמשים המחוברים

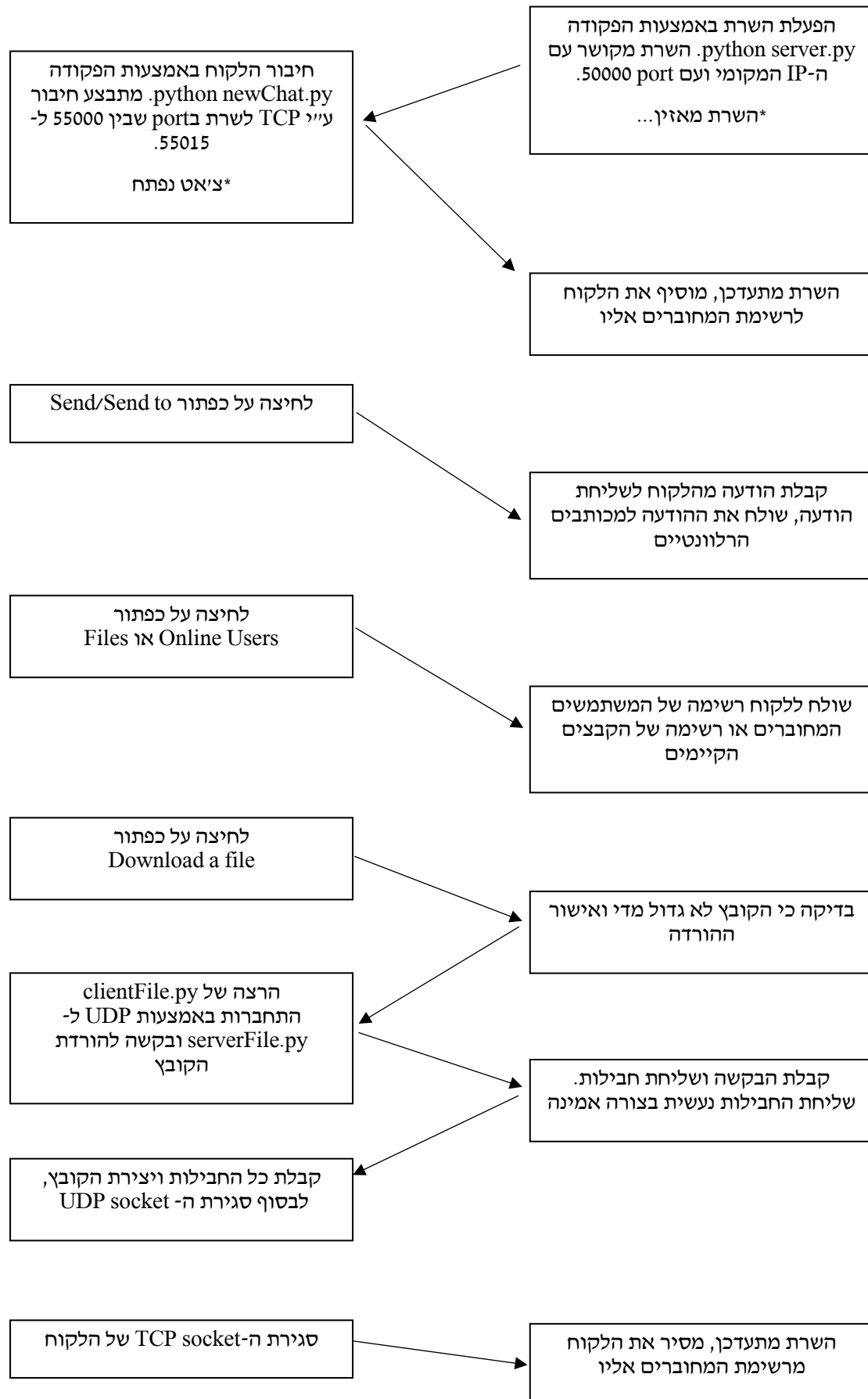


Send to – שליחת הודעה למשתמש יחיד *יש לכתוב את שם המשתמש בתיבה המתאימה



- כל לקוח יכול לעשות שימוש בכפתורים ללא תלות וללא השפעה על שאר המשתמשים, ניתן להשתמש בהם בו זמנית ותוך כדי שיחה.

דיאגרמת מצבים:



Wireshark:

	Info	Length	Protocol	Destination	Source	Time
13	Seq=1 Win=2619648 Len=4 [PSH, ACK] 55001 → 50000	48	TCP	127.0.0.1	127.0.0.1 15:48:03.588378	13
14	Ack=1 Win=327424 Len=0 [ACK] 50000 → 55001	44	TCP	127.0.0.1	127.0.0.1 15:48:03.588394	14
15	Ack=5 Win=327424 Len=4 [PSH, ACK] 50000 → 55001	48	TCP	127.0.0.1	127.0.0.1 15:48:06.738126	15
16	Ack=5 Win=2619648 Len=0 [ACK] 55001 → 50000	44	TCP	127.0.0.1	127.0.0.1 15:48:06.738150	16
17	Seq=6 Win=2619648 Len=19 [PSH, ACK] 55000 → 50000	63	TCP	127.0.0.1	127.0.0.1 15:48:06.738222	17
18	Ack=44 Win=327424 Len=0 [ACK] 50000 → 55001	44	TCP	127.0.0.1	127.0.0.1 15:48:06.738242	18
19	Seq=5 Win=2619648 Len=19 [PSH, ACK] 55001 → 50000	63	TCP	127.0.0.1	127.0.0.1 15:48:06.738259	19
20	Ack=24 Win=327424 Len=0 [ACK] 50000 → 55001	44	TCP	127.0.0.1	127.0.0.1 15:48:06.738280	20
21	Seq=24 Win=327424 Len=46 [PSH, ACK] 50000 → 55001	90	TCP	127.0.0.1	127.0.0.1 15:48:35.651261	21
22	Ack=51 Win=2619648 Len=0 [ACK] 55001 → 50000	44	TCP	127.0.0.1	127.0.0.1 15:48:35.651284	22
23	Seq=6 Win=2619648 Len=46 [PSH, ACK] 55000 → 50000	90	TCP	127.0.0.1	127.0.0.1 15:48:35.651359	23
24	Ack=90 Win=327168 Len=0 [ACK] 50000 → 55001	44	TCP	127.0.0.1	127.0.0.1 15:48:35.651372	24
25	Seq=51 Win=2619648 Len=46 [PSH, ACK] 55001 → 50000	90	TCP	127.0.0.1	127.0.0.1 15:48:35.651384	25
26	Ack=70 Win=327168 Len=0 [ACK] 50000 → 55001	44	TCP	127.0.0.1	127.0.0.1 15:48:35.651398	26

ניתן לראות כי מועברות חבילות מה-IP המקומי לעצמו, אך מ-port'ים שונים (50000 ו-55000).

נזכיר כי ה-port של השרת הוא 50000 וה-port של הלקוח הוא בין 55000 ל-55015, לכן ניתן להסיק מהצילום כי מועבר מידע בין השרת ללקוח. נראה לדוגמה כי באחת החבילות נמצאת הודעת ההתחברות שמועברת לכלל המחוברים לשרת כאשר משתמש נוסף מתחבר:

Wireshark · Packet 19 · Adapter for loopback traffic capture

19: 63 bytes on wire (504 bits), 63 bytes captured (504 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: 50000, Dst Port: 55001, Seq: 5, Ack: 5, Len: 19
 Data (19 bytes)
 Data: 2a2a204d656e69206a6f696e656421202a2a0a
 [Length: 19]

0000 02 00 00 00 45 00 00 3b aa 1a 40 00 80 06 00 00E..;..@.....
 0010 7f 00 00 01 7f 00 00 01 c3 50 d6 d9 bc 9f 72 61P....ra
 0020 46 b6 98 68 50 18 27 f9 39 e1 00 00 2a 2a 20 4d F..hP..'. 9...**M
 0030 65 6e 69 20 6a 6f 69 6e 65 64 21 20 2a 2a 0a eni join ed! **. ←

Bytes 44-62: Data (data.data)

עזרה סגור

נראה כי מני הצטרף לקבוצה ונשלחה הודעה לכלל המשתמשים ""* Meni joined! *""

בעת העברת קובץ נפתח UDP socket, ואכן נראה שכאשר משתמש הוריד קובץ כלשהו מהשרת הוא הועבר ב-UDP socket :

No.	Time	Source	Destination	Protocol	Length	Info
76	15:49:28.592239	127.0.0.1	127.0.0.1	UDP	33	Len=1 50000 → 55000
77	15:49:28.592278	127.0.0.1	127.0.0.1	UDP	37	Len=5 50000 → 55000
78	15:49:28.595321	127.0.0.1	127.0.0.1	UDP	37	Len=5 50000 → 55000
79	15:49:28.595463	127.0.0.1	127.0.0.1	UDP	38	Len=6 50000 → 55000
80	15:49:28.597799	127.0.0.1	127.0.0.1	UDP	217	Len=185 55000 → 50000
81	15:49:28.597991	127.0.0.1	127.0.0.1	UDP	38	Len=6 50000 → 55000
82	15:49:28.603486	127.0.0.1	127.0.0.1	UDP	217	Len=185 55000 → 50000
83	15:49:28.603623	127.0.0.1	127.0.0.1	UDP	38	Len=6 50000 → 55000
84	15:49:28.604675	127.0.0.1	127.0.0.1	UDP	217	Len=185 55000 → 50000
85	15:49:28.604778	127.0.0.1	127.0.0.1	UDP	38	Len=6 50000 → 55000
86	15:49:28.605161	127.0.0.1	127.0.0.1	UDP	138	Len=106 55000 → 50000
87	15:49:28.605236	127.0.0.1	127.0.0.1	UDP	38	Len=6 50000 → 55000
88	15:49:28.605573	127.0.0.1	127.0.0.1	UDP	117	Len=85 55000 → 50000
89	15:49:28.605631	127.0.0.1	127.0.0.1	UDP	38	Len=6 50000 → 55000

נפתח את אחת החבילות :

Wireshark · Packet 46 · Wireshark.pcapng

46: 37 bytes on wire (296 bits), 37 bytes captured (296 bits) on interface \Device\NPF_{Loopback}, id 0
 Null/Loopback
 Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
 User Datagram Protocol, Src Port: 55000, Dst Port: 50000
 Data (5 bytes)
 Data: 41434b2035
 [Length: 5]

0000 02 00 00 00 45 00 00 21 aa 88 00 00 80 11 00 00 ...E..!
 0010 7f 00 00 01 7f 00 00 01 d6 d8 c3 50 00 0d a6 44P...D
 0020 41 43 4b 20 35 ACK 5

Bytes 32-36: Data (data.data)

עזרה סגור

בחבילה זו הועברה הודעת ה-ACK 5' שהלקוח שולח לשרת כחלק מתהליך הורדת קובץ.

חלק ג'

שאלה 1 :

בהינתן מחשב חדש המתחבר לרשת אנא תארו את כל ההודעות שעוברות החל מהחיבור הראשוני ל switch ועד שההודעה מתקבלת בצד השני של הצ'אט. אנא פרטו לפי הפורמט הבא : סוג הודעה, פירוט הודעה והשדות הבאים: כתובת IP מקור/יעד, כתובת פורט מקור/יעד, כתובת MAC מקור/יעד, פרוטוקול שכבת התעבורה.

DHPC PROTOCOL

שלב א-

כשהמחשב שלנו החדש מתחבר הוא קודם כל צריך להקצות לעצמו כתובת IP משלו – לשם כך הוא מריץ את פרוטוקול DHCP המשמש לנתינת כתובות IP למחשבים חדשים. ההודעה נשלחת לכל הרשת המקומית – מה שמכונה צורת-BROADCAST

ההודעות יישלחו בפרוטוקול UDP

כל המחשבים ברשת המקומית מקבלים את ההודעה אך רק השרת DHCP מקבל אותה וכל היתר מתעלמים ממנה .

זה מתבצע כך : נשלחת הודעה בשם DHCP discovery :

כאשר : 0.0.0.0-IP.src

SRC.PORT הוא 68

DEST.PORT הוא 67

IP.DEST - 255.255.255.255

Src.MAC - כתובת MAC של המחשב המבקש .

MAC-dest ff-ff-ff-ff-ff-ff - כי מדובר בהודעת ברודקאסט ולכן המחשב לא יודע מה הוא הכרטיס רשת של HOST שהוא מחפש .

שלב ב-

כל שרת DHCP ברשת המקומית יקבל את ההודעה הזו וישלח הודעת DHCP OFFER בתמורה. מאחר ולמחשב החדש שלנו אין עדיין כתובת IP משלו – שרת DHCP יישלח את ההודעה גם כן לכל הרשת בצורת BROADCAST. תוכן ההודעה תהיה הצעה ל : IP adres , DNS server IP address , subnet mask , כתובת ה-IP של הנתב הראשי , משך הזמן שהשרת מקצה לו את הכתובת IP

מבנה ההודעה :

src.ip - זהו הכתובת של השרת .

ip.DEST - 255.255.255.255 - עדיין אין למחשב שלנו כתובת IP-

DEST.MAC - ff-ff-ff-ff-ff-ff

Src.MAC - כתובת MAC של השרת DHCP .

SRC.PORT - 67

PORT.DEST - 68

כל המחשבים ברשת המקומית מקבלים את ההודעה כי גם היא נשלחה ב broadcast אך ישנו שדה שנקרא client ID - שהוא ה MAC address של המחשב שלנו . כך שכל מחשב אחר ברשת שמקבל את ההודעה הזו יודע להתעלם ממנה.

שלב ג' - DHCP REQUEST .

המחשב שלנו שקיבל הצעה – שולח הודעת בקשה לקבל את את הכתובת אשר הוצעה כלומר ACK DHCP . אם ישנה רק הצעה אחת ומסיבה כלשהי הכתובת שהוצעה היא בעייתית אז נשלחת הודעה DHCPDECLINE שהיא בקשה לכתובת נוספת .

כתובת IP של המקור – 0.0.0.0 . כי עדיין אין לו כתובת Ip

כתובת ה-IP של היעד – 255.255.255.255

MAC.DEST - FF:FF:FF:FF:FF:FF.

MAC.SRC - מספק ה-MAC של הלקוח.

Src.port – 68

dest_port – 67

ההודעה מכילה שדה נוסף שנקרא Server Identifier - שם תיהיה כתובת ה ip של השרת שאיתו אנחנו מתקשרים . הסיבה שההודעה נשלחת בתפוצה מלאה ולא בצורה פרטנית לשרת הזה היא כדי שבשלב הבא – שרתים שיקבלו את ההודעהו ששלחו כתובת IP אך הלקוח בחר IP של שרת אחר – יקראו ויקראו את השדה Server Identifier שאינו זהה ל IP שלהם ואז : יאחסנו מחדש את הכתובת ה IP שהציעו .

שלב ד' - DHCP replay -

השרת מקבל DHCP REQUEST בתגובה הוא שולח DHCPACK שהוא למעשה אישור לכך שהתקבלה ה DHCP REQUEST והלקוח יכול להשתמש ב IP שהוצע לו . (במקרים מסוימים יישלח DHCPNACK שלמעשה אומר שההצעה כבר לא בתוקף .)

בנוסף הוא מעביר את כל התוכן שהוצע ללקוח בשלב ב' : subnet mask , DNS server IP , address , כתובת ה-IP של הנתב הראשי , משך הזמן שהשרת מקצה לו את הכתובת וכמובן ה IP שהוצע ללקוח .

מבנה ההודעה :

Src.IP – כתובת ה ip של השרת .

Dest.IP - 255:255:255:255

Src port - 67

Dest port - 68

MAC src - FF:FF:FF:FF:FF:FF

MAC dest - כתובת ה MAC של הלקוח

ARP PROTOCOL

כעת המחשב משתמש בפרוטוקול ARP :

ובdest_IP יהיה הכתובת של השרת DNS וdest_MAC תהיינה הכתובת של הנתב הראשי.

שאת שניהם קיבל במסגרת הפרוטוקול DHCP

במסגרת הפרוטוקול ARP נשלחת הודעת תפוצה לכל הרשת המקומית שבה נשאל למי יש את הכתובת MAC של הנתב הראשי – הנתב יקבל את ההודעה וישלח

Arp_replay עם כתובת MAC שלו .

ההודעה הזאת נשלחת בBROADCAST ולכן השדות הינם

IP.src - שהמחשב קיבל בפרוטוקול DHCP

255.255.255.255 זה הIP.DEST .

MAC -Src כתובת MAC של המחשב המבקש .

destMAC- ff-ff-ff-ff-ff-ff

שאלה 2 :

הסבירו מה זה CRC.

CRC - Cyclic Redundancy Check.

באמצעות CRC אנחנו מזהים שגיאות שנוצרו בעת העברת נתונים ברשת. ה-CRC מחושב ומקודד באמצעות מספר פעולות לפני שליחת ההודעה ומתווסף למידע המועבר, כאשר המידע מגיע לצד המקבל, הצד המקבל מבצע את אותן הפעולות, ובמידה וקיבל את אותה התוצאה הוא מאשר כי המידע הגיע אליו בהצלחה וללא שינויים.

שאלה 3 :

מה ההבדל בין HTTP 1.0, HTTP 1.1, HTTP 2.0, QUIC?

HTTP משתמש בפרוטוקול TCP שהוא connection-oriented, כלומר יוצר חיבור בהתחלה וסוגר אותו בסוף.

פרוטוקול HTTP 1.0 - פרוטוקול אשר פותח חיבור TCP לכל אובייקט בנפרד וסוגר אותו לאחר קבלת האובייקט. מה שאומר שאם הלקוח רוצה לקבל מספר אובייקטים מהשרת הוא יצטרך לפתוח מספר חיבורים. כלומר, חיבור אחד לכל אחד מהאובייקטים שהוא רוצה לקבל. לאחר מכן, הוא יסגור את החיבורים בנפרד. לכן, זמן התגובה הוא RTT לפתיחת חיבור TCP ראשוני + RTT לשליחת בקשת http להורדת אובייקט + זמן שידור האובייקט.

פרוטוקול HTTP 1.1 - פרוטוקול פתוח אשר נשאר פתוח לאורך כל הדרך, מוריד מספר אובייקטים וסוגר את התקשורת כאשר הוא מסתיים. שיטה זו משאירה את החיבור פתוח לאורך זמן, גם אם יש פער זמן בין שליחת האובייקטים על ידי השרת. לכן, זמן התגובה הוא RTT לשליחת בקשה לאובייקט מצד הלקוח, והוא מתקבל ע"י השרת כי החיבור כבר נוצר, אם זהו האובייקט השני המבוקש ע"י הלקוח + זמן שידור האובייקטים.

פרוטוקול HTTP 2.0 – מאפשר לשרת לדחוף אובייקטים ללקוח עוד לפני שהלקוח מבקש אותם. דחיפות השרת הן טכניקת ביצועים שמטרתה להפחית latency ע"י דחיפות אובייקטים ללקוח עוד לפני שהוא יודע שיהיה צורך בהם.

פרוטוקול Quick UDP Internet Connection - QUIC . כמשתמע משמו, הוא פרוטוקול מבוסס על UDP (בניגוד ל-HTTP) הנועד לאפשר חיבור מאובטח ומהיר בין המשתמש לבין האתר אליו מעוניין להגיע. גוגל פיתחה את QUIC כך שבמידה והמשתמש יצר קשר מאובטח מול השרת בעבר, שרת האתר יוכל לשלוח את המידע גם ללא בדיקת האתר והמתנה להקמת חיבור מאובטח.

שאלה 4 :

למה צריך מספרי port?

מספר port הוא מספר לוגי בגודל 16 סיביות המצורף לתהליך הפועל על המארח. הוא הכרחי מכיוון שכל מארח יכול להריץ מספר תהליכים במקביל בשליחת וקבלת נתונים, לכן יש לזהות את אותם התהליכים, כדי שהתהליך הרצוי ייקח את המידע מהsocket ולא תהליך אחר הרץ במקביל אליו, כלומר זה מאפשר לנו לבדל נתונים המיועדים לתהליכים שונים.

לסיכום, באמצעות כתובת ה-IP הנתונים מגיעים ל-host הנכון, אך הפורטים מבטיחים שהנתונים מגיעים לתהליך הנכון הפועל על אותו host, לכן השילוב בין השניים הכרחי.

שאלה 5 :

מה זה subnet ולמה צריך את זה?

subnet היא תת-רשת בתוך רשת, כתובת ה-IP מחולקת לוגית כך : החלק הראשון של הכתובת (ה-MSB bits) בכתובות ה-IP של מארחים המחוברים לאותה הרשת היא זהה, ומייצגת את כתובת הרשת אליה הם מחוברים. החלק האחרון של הכתובת מייצג את כתובת המארח הייחודית לכל מארח תחת אותה הרשת ולכן משתנה ממכשיר למכשיר, כלומר כתובת ה-IP של מארח בנויה משני חלקים : כתובת המארח וכתובת הרשת. (כמות הספרות הנדרשות לזיהוי ייחודי של כל מכשיר ומכשיר היא משתנה, כדי שאפשר יהיה להבחין בין חלק הרשת וחלק המארח בכתובת ה-IP, נהוג להצמיד לכל כתובת subnet mask, שמאפשרת באמצעות פעולה לוגית פשוטה למצוא את כתובת הרשת של כתובת IP מסוימת). החלוקה לתתי רשתות התבצעה מכמה סיבות ויתרונות : היא מגבירה את אבטחת הרשת - הרשת מוגדרת ע"י הנתבים, ה-routers, הם אלו שתוחמים לי את הרשתות, והם מהווים מעבר בין הרשתות השונות, הם מחליטים מי עובר ומי לא עובר. בנוסף, החלוקה נועדה על מנת לנהל את תעבורת הנתונים בצורה טובה יותר - בכך שאנו מגדירים אזורי שיחה, אזורים שניתן לתקשר ביניהם, לכן הדבר מפחית תעבורה מיותרת של הודעות וכן מפחית הודעות באופן כללי - הודעות שלא צריכות להגיע לכל מקום אלא רק לרשת מסוימת אחת. כמו כן, החלוקה לרשתות מאפשרת ניהול טוב יותר של הרשת, כך שאם מחלקות שונות יהיו חלק מרשתות שונות ניתן יהיה להגדיר להן תכונות שונות, ולכן הבעיות פוחתות ונעשות קלות יותר, וגם אם יש בעיות יהיה קל יותר לפתור אותן.

שאלה 6 :

למה צריך כתובות MAC למה לא מספיק לעבוד עם כתובות IP?

כתובת MAC היא מזהה ייחודי המוטבע על כל רכיב תקשורת בעת הייצור, כתובת ה-MAC מוטבעת בדרך כלל בכרטיס הרשת של המחשב ולא ניתנת לשינוי. לעומת זאת, כתובת ה-IP של מחשב משתנה מעת לעת.

כתובות MAC מטפלות בחיבור הפיזי ממחשב למחשב בלבד, בעוד שכתובות IP מטפלות בחיבור הלוגי הניתן לניתוב הן ממחשב למחשב והן מרשת לרשת. לכן נשאלת השאלה, מדוע אנו זקוקים גם לכתובות MAC?

נניח כי היו לנו כתובות MAC בלבד, ללא IP – צורה זו הייתה יעילה ברשתות קטנות או מקומיות, מכיוון שלא היה עולה הצורך לדעת מהו ה-IP של host מסוים, אלא רק את המזהה הייחודי שלו, אך מכיוון שרשת האינטרנט היא ענקית, צורה זו של תקשורת היא בלתי אפשרית, מכיוון שכתובות MAC אינן היררכיות, לא ניתן לנווט בעזרתן, לעומת כתובות ה-IP. בנוסף, אם כרטיס הרשת יפגע – כל החיבור הקיים יאבד.

כעת, נניח כי היו לנו כתובות IP בלבד, ללא MAC – בעזרת כתובות IP ניתן ליצור תתי רשתות, הן כתובות היררכיות ולכן ניתן לנווט בעזרתן. כתובת IP עוזרת לי לדעת היכן המארח, לפי הרשת בה הוא נמצא, אך לא מי הוא המארח.

בנוסף, אם מכשיר מתחבר לראשונה לרשת, הרי אין לו עוד כתובת IP ולא קיימים לו נתונים על השרתים הקרובים אליו. לכן הוא ישלח הודעת גילוי בצורת ברודקאסט על מנת למצוא את שרת ה-DHCP. כל עוד הלקוח לא קיבל כתובת IP הוא משתמש באפסים (0) על מנת לייצג את כתובתו ובינתיים הוא מזהה על פי כתובת ה-MAC שלו.

לסיכום, כתובת MAC אינה משתנה ולכן עוזרת לזהות מי הוא המכשיר, לעומת כתובת IP שהיא כתובת שיכולה להשתנות, ולכן עוזרת לזהות היכן המכשיר. כתובת MAC וכתובת IP יחד מנווטות את ההודעה להגיע למכשיר הנכון ברשת הנכונה. בנוסף, בעת חיבור ראשוני לרשת, כתובת ה-MAC מהווה כזיהוי ראשוני למכשיר, עד שהוא מקבל כתובת IP מהשרת.

שאלה 7:

מה ההבדל בין Router, Switch, NAT?

NAT – טכניקת ניתוב ברשת מחשבים, בה נכתבות מחדש כתובות ה-IP של חבילות שעוברות בנתב. כלומר, כתובת ה-IP שתצא משימוש תחזור אל מאגר הכתובות ותינתן למחשב אחר בשעת הצורך. NAT מאפשרת חיבור של מחשבים רבים הנמצאים באותה הרשת המקומית לרשת האינטרנט באמצעות כתובת IP אחת בלבד. יישום זה שימושי לצורך צמצום כתובות ה-IP בעולם, שהרי במקום שלכל מחשב תינתן כתובת IP חיצונית, כל המחשבים מיוצגים ככתובת אחת בלבד, וכן לשם חיבור לאינטרנט של רשת בעלת יותר ממחשב אחד, באמצעות חשבון אחד של חיבור מהיר לאינטרנט.

Router – רכיב תקשורת מחשבים שנועד לקביעת נתיבן והפצתן של חבילות נתונים ברשתות תקשורת נתונים. מחבר את הרשת הביתית לאינטרנט או לרשת אחרת חיצונית. בעת ניתוב מידע מרשת חיצונית למחשב או להתקן ברשת הפנימית, הראוטר משתמש בטבלת ה-NAT. תקשורת ברמת IP נקראת גם תקשורת בשכבת הרשת, שבו התעבורה עוברת ע"י כתובת IP.

Switch – כאשר התקשורת נעשית באותה הרשת, אין צורך בשימוש בראוטר, לכן נשתמש בסוויץ'. הסוויץ' הוא בעצם מעביר תקשורת בין רכיבי התקשורת באותה הרשת, תקשורת זו מבוצעת ע"י כתובת MAC address, שזוהי כתובת ייחודית לכל מכשיר, כפי שהסברנו בתשובה לשאלה הקודמת. לכל סוויץ' יש טבלה שבה יש שיוך בין פורט פיזי לבין כתובת MAC. תקשורת זו נקראת תקשורת ברמת שכבת התעבורה, שבו העברת המידע נעשית ע"י כתובת MAC.

לסיכום, תקשורת של התקני תקשורת ברשת פנימית מצריכה סוויץ' ולא ראוטר, התקשורת גם מהירה יותר כי אין צורך להמיר כתובות. כדי לתקשר עם רשת חיצונית או אינטרנט נהיה חייבים להשתמש בראוטר. הראוטר משתמש בטבלה שנקראת טבלת NAT כדי להמיר את כתובות ה-IP הפנימיים לכתובות IP חיצוניים.

שאלה 8 :

שיטות להתגברות על מחסור ב-IPv4.

ל-IPv4 שטח של 32 סיביות, (4 אוקטטות של 8 סיביות כל אחת) שטח זה אינו מספיק למספר הכתובות הנדרשות בתעשייה, לכן עלה הצורך לעבור לתקן עם כתובות נוספות – IPv6. ל-IPv6 שטח של 128 סיביות (8 אוקטטות של 16 סיביות) שזהו שטח עצום, 2^{128} כתובות שונות, הפותר את מצוקת כתובות ה-IP, כנראה לצמיתות.

בנוסף, דרך נוספת להתגבר על מחסור ב-IPv4 זה באמצעות NAT, שעליה פירטנו בשאלה הקודמת.

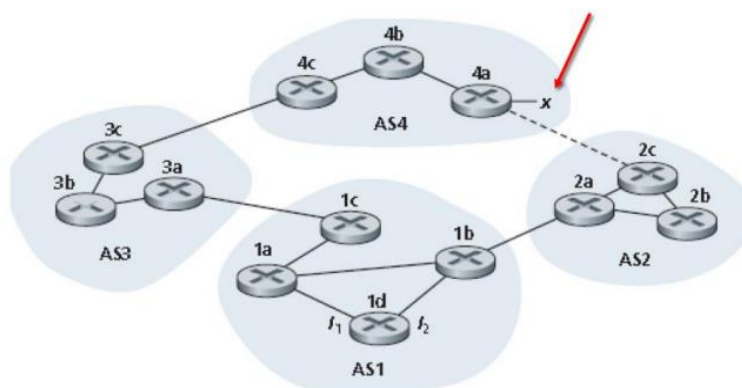
שאלה 9 :

נתונה הרשת הבאה:

a. AS2, AS3 מריצים OSPF

b. AS1, AS4 מריצים RIP

c. בין ASs רץ BGP



d. אין חיבור פיזי בין AS2, AS4

e. בעזרת איזה פרוטוקול לומד הנתב 3c על תת רשת x

f. בעזרת איזה פרוטוקול לומד הנתב 3a על תת רשת x

g. בעזרת איזה פרוטוקול לומד הנתב 1c על תת רשת x

h. בעזרת איזה פרוטוקול לומד הנתב 2c על תת רשת x

-E

בעזרת פרוטוקול RIP ילמד הנתב c4 על רשת x (כל נתב יתחזק טבלת מרחקים שבה לכל תת רשת רשום מספר הקפיצות בכדי להגיע אליה והנתב הבא שיש לקפוץ ממנו מהנתב)

הנתב c4 (נתב הגבול) יפיץ את המידע אודות הכתובות בתוך AS4 ובפרט – רשת x בעזרת פרוטוקול BGP שפועל בין ASs שונים הישר לc3 שהינו נתב הגבול של AS3

-F

בעזרת פרוטוקול RIP ילמד הנתב c4 על רשת x (כל נתב יתחזק טבלת מרחקים שבה לכל תת רשת רשום מספר הקפיצות בכדי להגיע אליה והנתב הבא שיש לקפוץ ממנו מהנתב)

הנתב c4 יפיץ את המידע לנתב C3 אודות x (בין היתר) בפרוטוקול BGP. לאחר מכן באמצעות פרוטוקול ASPF שפועל בין הנתבים בAS3, ילמד הנתב a3 על מצב הרשת x.

-g

כעת לאחר שא3 יודע את המידע אותות מצב רשת x , המידע יועבר מ3a לc1 באמצעות פרוטוקול BPG, שכן שניהם מהווים נתבי גבול של הAS שלהם .

-H

לאחר ש c1 למד את המידע אודות רשת x .

הוא יפיץ את המידע אודות הרשתות ב3as, 4as ובפרט הרשת x שבה אנחנו מעוניינים . זאת באמצעות פרוטוקול rip לשכנים שלו ב1as שיפיצו בעצמם לשכנים שלהם עד שהמידע יועבר לנתב b1. הנתב b1 מהווה נתב גבול של AS2 ולכן יפיץ את המידע באמצעות פרוטוקול BPG לנתב הגבול של AS2', הלא הוא a2 . הנתב a2 ימסור את המידע על x לנתב c2 באמצעות הפרוטוקול OSPF .

ביבליוגרפיה :

<https://tikshuv-ccna.com/7-1-0-%D7%A8%D7%99%D7%94-%D7%A2%D7%9C-%D7%97%D7%9C%D7%95%D7%A7%D7%94-%D7%9C%D7%A8%D7%A9%D7%AA%D7%95%D7%AA-%D7%AA%D7%AA%D7%99-%D7%A8%D7%A9%D7%AA%D7%95%D7%AA-%D7%95%D7%9B>

[/https://campus.gov.il/course/ariel-acd-computers-communication-he](https://campus.gov.il/course/ariel-acd-computers-communication-he)

https://he.wikipedia.org/wiki/%D7%9B%D7%AA%D7%95%D7%91%D7%AA_IP

<https://www.youtube.com/watch?v=oGoWqdlOMI>

<https://www.gadgety.co.il/128130/%D7%92%D7%95%D7%92%D7%9C-%D7%9E%D7%A7%D7%93%D7%9E%D7%AA-%D7%90%D7%AA-%D7%A4%D7%A8%D7%95%D7%98%D7%95%D7%A7%D7%95%D7%9C-%D7%94%D7%AA%D7%A7%D7%A9%D7%95%D7%A8%D7%AA-quic-%D7%A9%D7%99%D7%A1%D7%A4%D7%A7>

<https://www.dipole.co.il/support/%D7%A8%D7%90%D7%95%D7%98%D7%A8-%D7%90%D7%95-%D7%A1%D7%95%D7%95%D7%99%D7%A5-%D7%9E%D7%94-%D7%94%D7%94%D7%91%D7%93%D7%9C>

https://he.wikipedia.org/wiki/Network_Address_Translation#%D7%99%D7%AA%D7%A8%D7%95%D7%A0%D7%95%D7%AA_%D7%95%D7%97%D7%A1%D7%A8%D7%95%D7%A0%D7%95%D7%AA