בס''ד

**פרויקט סיום תקשורת ומחשוב -**  
**מגישים:** יוחנן קלינג – 316004555 צח אקר – 315339069

**חלק א':**הסבר השלבים:  
תחילה מריצים שרת על כתובת IP דיפולטיבית שלו כך שיוכל לקבל בקשות התחברות אליו דרך סוקט TCP, השרת מסוגל להאזין לעד 14 לקוחות בו זמנית.  
בעזרת THREADS, השרת מסוגל גם להקשיב לבקשות חיבור או קבלת הודעות TCP וגם לשלוח הודעות TCP ללקוחות, פרטי או הודעה לכולם.  
השרת מאזין על פורט ידוע מראש – 55000.  
בעזרת THREAD אחר, השרת מסוגל במקרה הצורך לשלוח הודעת UDP המכילה "שברירי קובץ" כך שבכל פקטה נשלח מידע מהקובץ בליווי מספר הפקטה ובכך מתמלא הבאפר ונשלח. מכיוון שהכל נעשה בתהליכונים שונים, השרת מסוגל גם לקבל וגם לשלוח מידע, הן בTCP והן בUDP.  
פורט הUDP השולח הוא 56000 ואילו הUDP המקבל הוא 57000, בכך הלקוחות שיודעים זאת מסוגלים לתקשר עם השרת בחיבורים ידועים מראש.  
  
לגביי הלקוח:  
כל לקוח תחילה מכניס את כתובת האיי פי של השרת, בהנחה והשרת יושב על כתובת IP דפולטיבית, ניתן להכניס איי פי ריק.  
בכך ניתן להתחבר לשרת עם כתובת ופורט כך שהלקוח מנסה לעשות BIND עם פורטים פנויים בסדר עולה מ55001 ועד 55002 עד שהוא מצליח להתחבר ואלו יהיו נתוני החיבור שלו.  
הלקוח בעזרת THREAD מאזין כל הזמן ומסוגל לשלוח הודעות לשרת או ללקוח אחר דרך השרת.  
על מנת להוריד קובץ הוא משתמש בTHREAD נוסף לחיבורי UDP כך שהוא יוכל להמשיך ולשמור על חיבור הTCP שלו פעיל תוך כדי.  
על מנת לשמור על חוקיות מסוימת בפורטים ולאפשר תקשורת מצד השרת, פורט היודיפי השולח הוא 200 יותר מפורט הTCP ואילו פורט המקבל הוא 100 יותר. בכך נוכל להבטיח סדר ושליטה על הפורטים ועל הכתובות הזמינות ודרכי התקשורת, כמובן השרת "יודע" זאת.

**חלק ב':**

**להלן דיאגרמת המצבים בהם המערכת עובדת:**

**שרת :**

Split file to package where size= buffer

Connect between socket to defined address

Ack++

**לקוח :**

Connect between socket to defined address

Ack++

**התגברות על איבוד חבילות:**

המערכת מתגברת על איבוד החבילות בכך שהיא ממתינה לתשובת "אוקיי" כל פעם על מנת להמשיך הלאה. כאשר אין תשובת "אוקיי", המערכת שולחת שוב את החבילה עד לקבלת "אוקיי", כך מנגנון זה מתגבר על איבוד החבילות.

**התגברות על בעיות latency:**

המערכת מתגברת על בעיות latency בכך שרק קליינט אחד יכול לדבר, באמצעות thread אחד שמאזין להודעות udp, thread אחר ששולח במקרה הצורך, thread אחר שמאזין לtcp ועוד thread אחר שאחראי על שליחת tcp.

כך שכל ה- threads פועלים "במקביל", למראית העין, שכן המחשב מבצע כל "תהליכון" למשך כמה שברירי שנייה, ובכך כאשר הודעה נשלחת, אם ה-thread שלה בפעילות, היא תשלח מיד ותתקבל כאשר thread הקבלה יחזור לפעילות ובינתיים תמתין ב"תור", ואם לא, ההודעה תישלח כאשר ה"תהליכון" שלה פועל ותתקבל כאשר "תהליכון" הקבלה יאשר אותה,

בכך ניצור מצב שרק הודעה אחת מלקוח אחד יכולה להתקבל בפרק זמן מסוים וכמעט במקביל אליו עובדים שאר ה-threads כך שלא ייווצר מצב בו הודעה "ממתינה" יותר מידי זמן, דבר אשר ידמה מצב שבו כמעט ואין השהייה, ונראה שהכל פועל במקביל וכל אחד יכול לדבר ולתזמן הודעות מתי שהוא רוצה.

**חלק ג':**

1. חיבור המחשב החדש צריך לקבל את כתובת ה- IP שלו, את כתובת הנתב, ואת כתובת שרת ה- DNS , נשתמש בפרוטוקול התקשורת DHCP.

בקשת ה- DHCP פועלת מעל פרוטוקול התעבורה UDP, מעל פרוטוקול הרשת IP ומעל פרוטוקול התקשורת אתרנט 802.3 (Ethernet).

שידור מסגרת האתרנט, יעד: FFFFFFFFFFFF ברשת LAN, התקבל בנתב שמריץ שרת ה- DHCP.

שרת ה- DHCP כותב DHCP ACK שמכיל את כתובת ה-IP של הלקוח, כתובת ה-IP של הנתב הראשון עבור הלקוח, שם וכתובת ה- IP של שרת ה- DNS.

לקוח DHCP מקבל תשובת DHCP ACK וכעת יש ללקוח כתובת IP, הוא יודע את השם והכתובת של שרת ה-DNS וכן יודע את כתובת ה-IP של הנתב הראשון שלו.

לפני שליחת בקשת HTTP, צריך את כתובת ה- IP של השרת DNS.

שאילתת DNS נוצרה פועלת מעל פרוטוקול התעבורה UDP, מעל פרוטוקול הרשת IP ומעל פרוטוקול התקשורת אתרנט.

כדי לשלוח מסגרת זו לנתב, צריך כתובת MAC של ממשק הנתב שהוא ARP.

שידור שאילתת ARP, מתקבל על ידי הנתב, שעונה עם תשובת ARP הנותנת את כתובת ה- MAC של ממשק הנתב.

הלקוח כעת יודע את כתובת ה-MAC של הנתב הראשון, ולכן יכול לשלוח כעת מסגרת המכילה את שאילתת ה- DNS.

שרת ה- DNS משיב ללקוח עם כתובת ה-IP של השרת.

כדי לשלוח בקשת HTTP, תחנת המקור שולחת הודעה לפתיחת קשר (TCP SYN). כעת, תחנת היעד מקבלת את ההודעה ושולחת בתגובה הודעת אישור קבלה ואישור פתיחת קשר מצידה TCP SYNACK. כעת נוצר חיבור TCP.

בקשת HTTP נשלחה לשקע TCP.

Datagram IP המכילה את בקשת ה- HTTP מנותבת אל שרת האינטרנט

הוא מגיב בתשובת HTTP, מכיל את דף האינטרנט, והיא (Datagram IP) מנותבת חזרה ללקוח.

1. CRC (Cyclic redundancy check) זו שיטת בדיקת תקינות המסר שעבר בין מחשבים כך שלאחר ביצוע בדיקת התקינות הלקוח והשרת יודעים שהמסר הועבר בהצלחה.  
   בדומה לchecksum שיטת הCRC מוסרת מידע נוסף מעבר להודעה עצמה ופרטי ההתקשרות כך שהמחשב המקבל יוכל לוודא שההודעה שהוא קיבל אכן זו ההודעה שנשלחה.
2. ההבדל העיקרי בין גרסת 1.0 לגרסת 1.1 הוא שבגרסה 1.0 בקשות הGET לקבלת מידע נשלחות אחת אחת ועד שאין תשובה על הGET הנוכחי, לא נבקש את האובייקט הבא אחריו ובכל בקשה נפתח קשר TCP מחדש  
   (non persistent connection)  
    לעומת גרסה 1.1 בה כלל האובייקטים מבוקשים באותה פתיחת קשר TCP (עד לגבול המתאפשר באותו חיבור HTTP) כך שנסגר קשר הTCP רק בסוף הקבלה (persistent connection).   
   בגרסה 2 החידוש הוא שבאותו חיבור TCP מבוקשים במקביל כל האובייקטים הרצויים והם יכולים להגיע במקביל.  
   בחיבור מסוג QUIC החיבור מבוצע לא על ידי TCP אלא על ידי UDP כך שהוא מהיר יותר שכן לא מחכים לתשובת הTCP  
   בגרסה זו גם החיבור אמין יותר.
3. port הוא שער המזהה את האפליקציה באופן ייחודי על אותו המחשב, כמין הדרכה למחשב השולח והמקבל – לאן להעביר את המידע שישלח או שהתקבל.
4. סאבנט זו הרשת הפנימית אליה מחשב מחובר. למשל, חברה, עסק, אוניברסיטה – לכל המחשבים המחוברים לאותה הרשת יש תחילת IP משותפת כדי שהמחשב ידע שהיעד נמצא בסאבנט מסוים, ולאחר שהוא מגיע לסאבנט מסוים, הוא מחפש את סיומת הIP שמצביעה למחשב הספציפי הקיים באותה הרשת הפנימית המחוברת בעזרת הSWITCH והראוטר.
5. כתובת הMAC היא כתובת צרובה שלא ניתנת לשינוי והיא ייחודית לכרטיס הרשת הספציפי הזה כך שאין עוד אחת הדומה לה. בניגוד לכתובת IP שהיא כתובת לוגית ולא פיזית הניתנת לשינוי.  
   כתובת MAC היא זו האומרת לראוטר מי הוא המחשב המבקש שירות ולאחר מכן הוא מקבל IP.  
   כך שללא כתובת הMAC לא היינו יכולים לזהות בצורה ייחודית מחשב ברשת ולשייך לו כתובת IP חדשה שכן אם היינו מתבססים רק על כתובת IP שכל הזמן מתחדשת ומשתנה בהתאם לתנאי מיקום הסכמי רשת ומוסכמות כאלו ואחרות, לא היינו יכולים לזהות בצורה וודאית מחשב מאוד ספציפי ברשת כך שכאשר תתקבל פקטה והראוטר יעביר בשידור לכולם, רק המחשב עם הMAC הספציפי ירצה לקבל את המסר.
6. NAT הוא חיבור מחשבים רבים הנמצאים באותה הרשת המקומית לרשת האינטרנט באמצעות [כתובת IP](https://he.wikipedia.org/wiki/%D7%9B%D7%AA%D7%95%D7%91%D7%AA_IP) אחת בלבד כך שגם כתובת המחשב הספציפית אינה גלויה לכולם וגם כך אנו חוסכים בכתובות IP שכן ישנה הכתובת הציבורית המגיעה אל הראוטר המתאים ולאחר מכן הראוטר משדר למי שמחובר אליו באמצעות הכתובת הפרטית שלו.  
   ROUTER – התקן בשכבת הרשת, תפקידו לנתב הודעות לכתובות IP חיצוניות ולקבל הודעות ולנתב אותן לכתובת המבוקשת הפנימית בעזרת שמירת והמרת כתובות IP פנימיות לחיצונית ( ובכך לחסוך בכתובות IP מוקצות).  
   SWITCH – מעביר מידע בתוך הרשת הפנימית בעזרת כתובת MAC ADRESS ללא צורך בכתובת IP חיצונית וללא צורך הכרחי להיעזר בנתב "ולצאת" החוצה אל הרשת החיצונית.
7. IPV4 מיוצגת בעזרת 32 ביטים המייצגים את כתובת המחשב, כך שישנם כ4.3 מיליארד אפשרויות לכתובות IP שונות, כדי להתמודד עם מחסור כתובות הIP הזמינות פותחה שיטת פרוטוקול IP חדשה בשם IPV6 כך שהכתובת מיוצגת על ידי 128 ביטים, מספר עצום כך שמחסור הכתובות לא יורגש.  
   בנוסף ניתן לחסוך בכתובות IP באמצעות הראוטר כך שיכולות להיות 2 כתובות IP פנימיות דומות ב2 רשתות שונות אך ישנה כתובת IP אחת חיצונית לכל רשת ובכך ניתן ליצור כפילויות בכתובות הIP הפנימיות.
8. e. הנתב 3c לומד על תת רשת x בעזרת פרוטוקולים BGP, RIP .

שהרי נשתמש בפרוטוקול הניתוב BGP במעבר מ- AS3 ל- AS4 ופרוטוקול זה בעצם פועל על פרוטוקול TCP. כעת נשתמש בפרוטוקול הניתוב RIP כדי להגיע ל-x מתוך AS4, ופרוטוקול זה בעצם פועל על פרוטוקול UDP.

f. הנתב3a לומד על תת רשת x בעזרת פרוטוקולים BGP, RIP, OSPF.

שהרי נשתמש בפרוטוקול הניתוב OSPF במעבר מ- 3a ל- 3c . כעת נשתמש בפרוטוקול הניתוב BGP במעבר מ- AS3 ל- ופרוטוקול זה בעצם פועל על פרוטוקול TCP. כעת נשתמש בפרוטוקול הניתוב RIP כדי להגיע ל-x מתוך AS4, ופרוטוקול זה בעצם פועל על פרוטוקול UDP.

g. הנתב1c לומד על תת רשת x בעזרת פרוטוקולים BGP, RIP, OSPF.

שהרי נשתמש בפרוטוקול הניתוב BGP במעבר מ- AS1 ל- , AS3 נגיע ל- 3a ופרוטוקול זה בעצם פועל על פרוטוקול TCP. כעת נשתמש בפרוטוקול הניתוב OSPF במעבר מ- 3a ל- 3c . כעת נשתמש בפרוטוקול הניתוב BGP במעבר מ- AS3 ל- ופרוטוקול זה בעצם פועל על פרוטוקול TCP. כעת נשתמש בפרוטוקול הניתוב RIP כדי להגיע ל-x מתוך AS4, ופרוטוקול זה בעצם פועל על פרוטוקול UDP.

h. הנתב 2c לומד על תת רשת x בעזרת פרוטוקולים BGP, RIP, OSPF.

שהרי נשתמש בפרוטוקול הניתוב OSPF במעבר מ- 2c ל- 2a . כעת נשתמש בפרוטוקול הניתוב BGP במעבר מ- AS2 ל- AS1(מ- 2a ל- 1b). כעת נשתמש בפרוטוקול הניתוב RIP כדי להגיע ל-1c מתוך AS1. כעת נשתמש בפרוטוקול הניתוב BGP במעבר מ- AS1 ל- , AS3 נגיע ל- 3a ופרוטוקול זה בעצם פועל על פרוטוקול TCP. כעת נשתמש בפרוטוקול הניתוב OSPF במעבר מ- 3a ל- 3c . כעת נשתמש בפרוטוקול הניתוב BGP במעבר מ- AS3 ל- ופרוטוקול זה בעצם פועל על פרוטוקול TCP. כעת נשתמש בפרוטוקול הניתוב RIP כדי להגיע ל-x מתוך AS4, ופרוטוקול זה בעצם פועל על פרוטוקול UDP.