שאלות נוספות פרוייקט גמר:

1. הבדלים בין TCP ו- QUIC:
   1. חיבור TCP דורש 3-way handshake ז"א הלקוח שולח הודעת חיבור לשרת, השרת שולח קיבלתי עם הודעת חיבור ואז הלקוח שולח אישור על ההודעת חיבור של השרת. QUIC לעומת זאת, משתמש ב RTT שמשמעותו חיבור יותר מהיר ומפחית עומס (חיבור בעשרת הודעה אחת בלבד).
   2. QUIC מאפשר שליחת מספר הודעות לחיבורים שונים במקביל דרך אותו חיבור בעוד TCP לא...
   3. Quic מוצפן by default. בעוד tcp אינו
   4. כאשר חבילה לא הגיעה, או הייתה בעיה (למשל ב checksum) TCP שולח מחדש את אותה פקטה בעוד QUIC יצרף את הדאטה החסרה לפקטה הבאה
2. Cubie ו vegas:
   1. Vegas מקבל פידבק מהרשת, בעוד cubie מתחשב במספר ההודעות באוויר לעומת גודל החלון
   2. לvegas התכנסות איטית יותר לעומת cubic. מה שהופך אותו לפחות מדוייק מכיוון ונדרשות יותר פעולות
3. BGP הינו פרוטוקול ניתובת בו כל ראוטר מעביר אינפורמציה על מסלול. הוא אינו מעביר את מחיר המסלול ואיזה מסלול עדיף, החלטה זאת נשארת לשיקול האדמיניסטרטור של כל רשת. ההבדל הוא שכל ASs שומר לעצמו מי השכנים שלו (למי הוא יכול להעביר מידע) מה שלא נקבע לאו דווקא על פי קשר פיזי ממש. ככה המידע שמקבל ה admin של הרשת זה איזה ASs יש לו בדרך ליעד ולמי כל אחד מקושר. ככה הוא יכול לבחור בעצמו את הדרך הרצויה (על פי פרמטרים שהוא מחליט). שונה מ OSPF בכך שבכל ראוטר שמור המידע שלו עם השכנים שלו, בניגוד לOSPF שכל המידע בין הראוטרים משותף וככה גם אמין יותר. אי אפשר ל"רמות". OSPF בהגדרה עובד לפי דיאקסטרה בעוד BGP אנו יכולים לבחור בעצמנו את הדרך העדיפה עלינו לפי שיקולינו. לכן BGP לאו דווקא עובד לפי מסלולים קצרים, אפשר להכניס שיקולים נוספים.
4. DNS משמש לתרגום שמות של דומיינים (URL) לכתובות IP (משמש בעיקר בWLN) בעוד ARP משמש לתרגום כתובות IP לכתובות mac ברשת פנימית. בפרוטוקול DNS מוחזק מאגר של כתבות ואנו שואלים את הserver המתאים לקבלת IP לפי מפתח. ARP לעומת זאת אנו שוולחים ברשת הפנימית למי שייך כתובת IP מסויימת, והמכשיר הרלוונטי מחזיר לנו את הכתובת mac שלו. לכן ההבדלים הם ב: איזה ערך אנו מחפשים (IP או mac) ובדרך קבלת המידע (ממאגר או מהמכשיר הרלוונטי) ובזה שאחד לרשת חיצונית לעומת רשת פנימית

איבוד חבילות של המערכת:

כאשר נשלחת הודעה (redirect), המערכת מחכה מאית שניה לקבלת ack. אם לא הגיעה, נשלך שוב, ככה חמש פעמים. אם אחרי חמש שליחות לא נקבל תשובה, נוותר. נשים לב שההמתנה מתבצעת ב thread נפרד כאשר הthread הראשי הוא זה שמקבל הודעות מהבאפר.

ככה אנו מתגברים על איבוד חבילות (לא קיבלנו ack, נשלח עוד פעם) וlatency בכך שחיכינו, שלחנו כמה פעמים (עם השהייה) ולכן גם אם לקח זמן עד שהגיעה החבילה, בינתיים נשלח עוד פעם ויכול להיות שנקבל ack על הודעה קודמת (אותה הודעה רק שליחה אחרת).

בנוסף אם השרת רואה שאנו לא מצליחים לעבוד עם כל הלקוחות, הוא מוריד את מספר הלקוחות איתם הוא עובד עד שיצליח להתמודד ואז מעלה את המספר בהתאם.

ישנה פונקצייה send all בUDP אשר שולחת

דיאגרמת מצבים של המערכת:

Send back redirect

Wait for ack

Make 3-way handshake

Send back redirect

New connection

GET request

listening

GET request

Unknown connection

listening

start

TCP server

Reliable UDP server