פרויקט סוף QUIC – חלק יבש

1. 5 מגבלות של פרוטוקול TCP:

א. **חיבור לא יציב כסביבה משתנה** – בעת שינוי חיבור (למשל שינוי כתובת IP כתוצאה משינוי מיקום/חיבור) TCP אינו מצליח להסתגל לשינוי. כתוצאה מכך החיבור ינותק וכל המידע שהועבר יאבד.

ב. **אובדן חבילות גורר עיכוב משמעותי** – מגבלה הידועה בשם "חסימת ראש התור". ב-TCP החבילות מתקבלות בשכבת האפליקציה בזו אחר זו, כאשר יש חשיבות לסדר. כאשר חבילה מתעכבת או נאבדת, זה יגרור עיכובים בחבילות שיגיעו לאחר מכן, כיוון שהן לא יעברו לשכבת האפליקציה עד אשר החבילה שאבדה תשודר ותתקבל.

ג. **תקורה גבוהה במידע המועבר –** ב-TCP יש מנגנון של קבלת אישורים (ACK) על כל הודעה שהועברה. זה מוסיף תקורה של מידע לא שימושי שעובר במהלך החיבור בין הצדדים, בכמות גבוהה. למרות שקיימת ב-TCP תמיכה חלקית ב-Selective Ack, מדובר בתמיכה מוגבלת בשל מבנה ההודעה של TCP (יש מגבלה על הטווחים בכותרת של TCP), ולכן התמיכה אינה מלאה.

ד. **עיכובים הנובעים מבקרת העומס –** למרות שתפקיד בקרת העומס הוא למנוע גודש ברשת, חלון השליחה עשוי לייצר עיכובים משמעותיים. כל אובדן חבילה עוצר את התקדמות החלון עד אשר החבילה מתקבלת, על אף שחבילות אחרות מחלון השליחה כבר כן הגיעו ליעד. כלומר יש תלות בחבילה בודדת בהתקדמות חלון השליחה.

ה. **תהליך לחיצת היד אינו משלב הגדרת חיבור מאובטח –** יש צורך בתהליך נוסף, מעבר ללחיצת היד הרגילה בין 2 נקודות הקצה, ורק לאחר התהליך (שכולל החלפת מזהים קריפטוגרפים ואישורי אבטחה) תחל העברת המידע. כלומר יש צורך בשני סבבים של לחיצת יד.

2. <u>תפקידים שפרוטוקול תעבורה צריך למלא:</u>

א. בקרה על עומס - בקרה על כמות המידע (חבילות) העוברות בין 2 החיבורים, על מנת שהיעד יוכל לעמוד בעומס הנתונים מבחינת קיבולת הקו.

ב. אבטחת מידע – הקמת חיבור מאובטח בין 2 הצדדים, לשם מסירת מידע מהימן תוך אימות הצד השני.

ג. תחזוקת החיבור - יש לנהל את החיבור באופן שיאפשר לבסס את החיבור בין 2 הצדדים, העברת מידע וניתוק בתום הקשר.

ד. ניהול רצף נתונים – הנתונים צריכים להגיע בסדר הנכון ולאפשר הרכבת המידע בסדר זה, גם אם בפועל נקלטו בסדר שונה.

ה. זיהוי עומסים ברשת ו-ויסות בקצב השידור בהתאם – ניתוח זמן תגובה וזיהוי מצב של עומסים ברשת על מנת להימנע מאיבוד חבילות או עיכובים.

3. <u>תהליך פיתחת הקשר ב-QUIC</u>

כדי לייצר פרוטוקול יעיל, יש ב-QUIC מנגנון של התחברות חוזרת על מנת לחסוך את החלפת הסוד. נציג את החיבור הרגיל ולאחר מכן את החיבור החוזר (Retry)

<u>חיבור רגיל</u>

1. הלקוח יוזם את התהליך ע"י שליחת הודעת "Client Initial" לשרת. הוא יצרף להודעה מספר זיהוי של החיבור, ונתונים קריפטוגרפיים (לשם אימות).
2. השרת ישלח בתגובה הודעת "Server Initial" בה יצרף גם הוא מידע קריפטוגרפי, ואת מספר החיבור שלו.
3. הלקוח מקבל את ההודעה, ישלח ACK ומבצע תהליך אימות עם השרת (ייתכן במספר הודעות).
4. הלקוח יעביר לשרת הודעה בתום התהליך בה יודיע לו כי תהליך האימות הושלם.
5. השרת שולח אישור שתהליך האימות מבחינתו הושלם וניתן להתחיל להעביר הודעות ע"י מפתחות ההצפנה שהועברו.

<u>חיבור חוזר</u>

1. הלקוח יוזם את התהליך ע"י שליחת הודעת "Client Initial" לשרת. הוא יצרף להודעה מספר זיהוי של החיבור, ונתונים קריפטוגרפיים (לשם אימות).
2. השרת ישלח בתגובה הודעת Retry, בה יוודא את הלקוח. הודעה זו מכילה מספר חיבור חדש, ו-token ייחודי שנוצר ע"י השרת. בנוסף מועבר מידע קריפגטורפי כמו קודם.
3. הלקוח מקבל את ההודעה, וישלח מחדש את ההודעת "Client Initial" שכוללת את ה-token שהתקבל מהשרת. ההודעה מכילה גם מידע נוסף, שהיה דרוש לשרת על מנת לאמת שמדובר בלקוח מאומת.
4. השרת יקבל את ההודעת "Client Initial" החדשה ויבדוק את הנתונים שהתקבלו. לאחר שוויודא כי הפרטים נכונים, יחזיר הודעת "Server Initial" בה יצרף מידע נוסף לשם השלמת התהליך.
5. הלקוח מקבל את ההודעה, ישלח ACK ומבצע תהליך אימות עם השרת (ייתכן במספר הודעות).
6. הלקוח יעביר לשרת הודעה בתום התהליך בה יודיע לו כי תהליך האימות הושלם.
7. השרת שולח אישור שתהליך האימות מבחינתו הושלם וניתן להתחיל להעביר הודעות ע"י מפתחות ההצפנה שהועברו.

יתרונות ביחס ל-TCP:

1. תהליך קצר יותר – תהליך לחיצת הידיים משתמש ב-RTT בודד להקמת הקשר, בעוד ב-TCP יש את הקמת החיבור הבסיסי ולאחר מכן את הקמת הערוץ המאובטח באמצעות TLS.

2. תמיכה בחיבור חוזר – אם הלקוח כבר תקשר עם השרת ומחזיק בסוד משותף איתו, הקמת הקשר תיקח סיבוב (RTT) אחד פחות.

3. האבטחה מובנית בפרוטוקול ולא מצריכה תהליך אימות בנוסף כפי שיש ב-TCP.

4. עמידות לשינויים ברשת - מספר הזיהוי שמוחלף בחיבור אמור לאפשר לשני הצדדים לתקשר גם כאשר יש שינויים אצל אחד מהם. מה שמאפשר לחיבור להיות יציב. ב-TCP החיבור היה מנותק כאשר ה-IP היה משתנה.

4. מבנה חבילת QUIC

הפרוטוקול מגדיר 2 סוגי חבילות:

א. **חבילות בעלות כותרת ארוכה (Long Header Packets)** - חבילה לביסוס חיבור. משמשת להעברת נתוני חיבור ראשוניים ולחיצת ידיים מוצפנת של TLS.

מבנה:

- סוג החבילה (Packet Type) - מציין את סוג החבילה
- גרסת הפרוטוקול
- מספר זיהוי של חיבור המקור (Source Connection ID)
- מספר זיהוי חיבור יעד(Destination Connection ID)
- אורך כותרת (Length Field)
- מספר רצף(Packet Number)
- דגלים – Initial, Retry, Handshake, RTT-0
- המידע בפועל (Payload)

מבנה זה תוכנן כדי להתמודד עם מספר חסרונות שצוינו מעלה של פרוטוקול TCP:

1. תהליך לחיצת ידיים קצר יותר – החבילה מכילה נתונים שנדרשים לאימות והחלפת מידע קריפטוגרפי, מה שהופך את תהליך לחיצת הידיים לקצר בהשוואה ל-TCP.

2. אבטחה מובנית – TCP אינו מגיע עם הצפנה כברירת מחדל ונאלץ לבצע את תהליך ההצפנה בנפרד.ב-QUIC ההצפנה היא מובנית ומבנה החבילה מאפשר זאת באמצעות הדגלים המצורפים בכותרת (header).

3. שמירה על מזהה חיבור - TCP לא עמיד בשינויים בכתובות IP. מבנה הכותרת של QUIC מאפשר ניהול חיבורים גם בסביבת רשת משתנה, הודות למספר הזיהוי של החיבור.

ב. **חבילות עם כותרת קצרה (Short Header Packets)** - מיועדות להעברת המידע ולתקשורת לאחר שהחיבור בין 2 הצדדים הושלם.

מבנה:

- סוג החבילה (Packet Type) - מציין את סוג החבילה
- דגלים
- מספר זיהוי חיבור יעד(Destination Connection ID)
- מספר מזהה ייחודי של החבילה (מוצפן)
- המידע בפועל (Payload), **בחבילות של ACK הוא מכיל את הטווחים של האישורים הסלקטיביים:**
    - Largest Acknowledged- מספר החבילה הגבוה ביותר שהתקבל בהצלחה.
    - ACK Ranges - רשימה של טווחים של חבילות שהתקבלו בהצלחה

מבנה זה תוכנן כדי להתמודד עם מספר חסרונות שצוינו מעלה של פרוטוקול TCP:

1. הפחתת התקורה – השימוש בכותרת קצרה מפחית את המידע הלא שימושי שנשלח. כמו כן יש פחות צורך בשימוש ב-ACK לכל הודעה הודות למנגנון שבו כל האישורים נשלחים בצורה של טווח.

2. מענה לבעיית "חסימת ראש התור" – העובדה כי האישורים הם סלקטיביים מאפשרת בהמשך התקדמות שליחת המידע גם אם חבילה מסוימת אבדה, בניגוד ל-TCP שנאלץ להתעכב עד שהחבילה תתקבל.

5. כאשר חבילה הנשלחת מתעכבת/לא מגיע בזמן, הצד השולח מזהה זאת בפריים של ACK בו החבילה לא נכללה בטווח. הצד השולח יגיב בשידור חוזר מהיר (Fast Retransmission) במטרה לשמור על התקשורת יציבה, כמנגנון להתאוששות מהירה. במידה ומתברר כי אובדן החבילה מגיע כתוצאה מגודש ברשת, אלגוריתם בקרת העומס יוריד את קצב השידור כדי להתאים אותו אל לקצב התעבורה שהרשת יכולה להתמודד.

6. <u>בקרת העומס ב-QUIC (NewReno)</u>

פרוטוקול NewReno מנהל את קצב שליחת הנתונים כך שהביצועים יהיו מיטביים.

עקרונות שיש ב- NewReno ב-QUIC קיימים גם ב-TCP, אך ב-QUIC קיימים שדרוגים הקשורים למדידת זמני RTT העוזרים לאלגוריתם להבין את מצב הרשת.

בכל פעם הוא מבצע הערכת זמן להשהיה ומעריך את ה-RTT הבא, בכך הוא מתאים באופן דינמי את זמן ה-timeout של כל חבילה. אם מזוהה עליה ב-RTT, QUIC יכול להקטין את חלון הגודש ולהתאים את קצב השידור, על מנת למנוע עומסים.

המצבים בבקרת העומס של QUIC:

1. התחלה איטית – קצב השידור עולה באופן אקספוננציאלי. שלב זה נועד לבחון את היכולת של הרשת להתמודד, כך שהמטרה היא למקסם את משלוח החבילות. מצב זה הוא התחלתי ועשויים לחזור אליו כאשר יש גודש מתמשך ברשת (Persistent Congestion). כשיש חיווי הגעה ACK גודל החלון גדל, כשיש איבוד - נצא ממצב זה.

2. שחזור – כאשר יש אובדן של חבילה, או אינדיקציה לגודש ברשת, חלון העומס יורד בחצי וסף ההתחלה האיטית SST מתעדכן.

3. מניעת עומס – בניגוד למצב של ההתחלה האיטית, הקצב עולה באופן לינארי. המטרה היא להגדיל את החלון בהדרגה ולהימנע מעומס נוסף.