בעלי הפרויקט:

דני איסקוב, 206530073
קורן עבדוש, 209271535
יהונתן ברוך, 211465786

קישור לקוד:

https://github.com/yonibaru/networks_finale

החלק ה"יבש" בפרויקט

5 חסרונות/מגבלות של TCP

1. חוסר יעילות שנובע משימוש באותו אלגוריתם עבור Congestion Control וגם Reliable Delivery

TCP שואף לשלוט במדויק במספר הפקטות שנמצאות בתוך הרשת ומסתמן שהוא משתמש באותו האלגוריתם חלון עבור flow control וגם reliable delivery שזה לא אופטימלי - לדוגמא, אם פקטה הולכת לאיבוד אז חלון שלם של פקטות "ייזרק" אפילו אם חלק מהם כבר הגיעו ליעד בהצלחה.

2. Head of Line Blocking - HoL

TCP מוודא שהדאטא מדולבר בסדר מסוים, אך אם פקטה אחת הולכת לאיבוד, אז כל הפקטות שאחריה יחכו עד שהפקטה שהלכה לאיבוד תשלח שוב ותתקבל ובכך להאט את התפוקה של ההעברה.

3. חיבור ראשוני איטי

לפני שTCP יכול לשלוח דאטא, חייבת להתקיים לחיצת יד משולשת (3-way handshake), תהליך שלוקח זמן. הדילי הזה יכול להוות בעיה, במיוחד שדרוש חיבור מהיר. בנוסף, פרוטוקולים מתקדמים של TCP יממשו גם תהליכי אימות ואבטחה לפני החיבור, מה שאפילו יאט את החיבור הראשוני עוד יותר.
(לדוגמא בעת שימוש באינטרנט כנראה שנצטרך גם לבצע לחיצת יד TLS על מנת להבטיח חיבור מאובטח)

4. Header קבוע בגודלו

לTCP יש Header בגודל קבוע, מה שמגביל כמה עם כמה מידע הוא יכול לעבוד בצורה יעילה, במיוחד במהירויות גבוהות.

5. שינוי כתובת IP

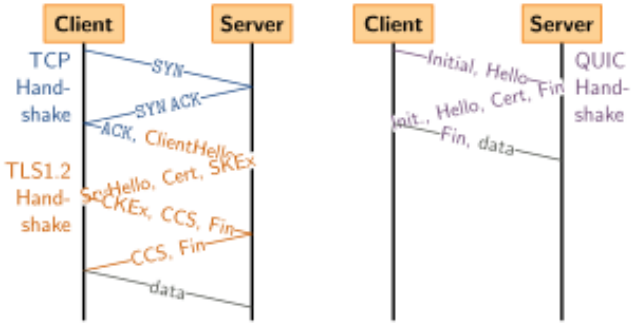
חיבור TCP מאופיין ע"י כתובת IP ומס' הפורט הרלוונטי. אם כתובת הIP תשתנה בעת חיבור, נצטרך להתחבר מחדש - מה שגורר handshake חדש וכפי שאמרנו ב3. זה תהליך שהוא איטי.

5 תפקידים שפרוטוקול תקשורת צריך למלא

1. Reliability: הפרוטוקול צריך להבטיח שהנתונים אכן מועברים ויוכל לשדר מחדש נתונים שאבדו.
2. Efficiency: הזמן שלוקח לעיבוד והעברת נתונים צריך להיות מהיר יחסית.
3. Scalability: הפרוטוקול צריך לתמוך בגדלי רשת שונים, במיוחד בכמות גדולה.
3. Compatibility: הפרוטוקול צריך לעבוד על פלטפורמות רבות ללא בעיה דבר שהוא חשוב במיוחד בעידן של היום.
4. Secure: הפרוטוקול צריך להיות עמיד מפני גישה לא רצויה ולהגן על הנתונים שלו באופן מסוים (הצפנה).
5. Adaptive: הפרוטוקול אמור להיות מסוגל להתאים את עצמו לתנאי הרשת על מנת למקסם את הביצועים שלו.

לחיצת היד כפי שהיא ממומשת בQUIC

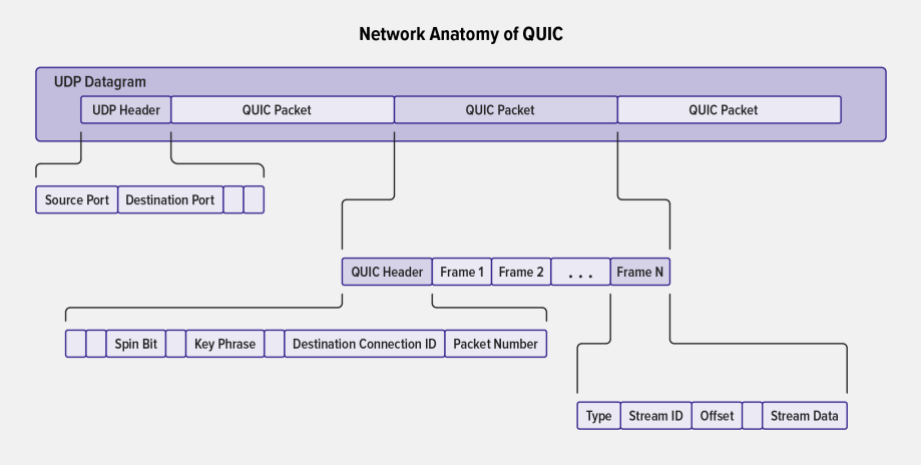
QUIC משלבת את הלחיצת היד התעבורתיות והקריפטוגרפיות (לדוגמא TLS) אל תוך תהליך יחיד שלוקח 1-RTT בלבד. בנוסף, לחיצת היד של QUIC משלבת הצפנה (דרך TLS 1.3) דרך הפרמטרים התעבורתיים. התהליך הזה הרבה יותר מהיר בהשוואה לTCP שעושה את תהליך לחיצת היד ב2-RTT. בQUIC, הפקטה הראשונית מנהלת "משא ומתן" על מזהי החיבור ובנוסף היא גם מהווה לחיצת יד של TLS על מנת לאבטח פקטות עתידיות. השרת יכול לאמת את הכתובת הלוקח עם פקטה נוספת אך זה לא חובה.
בנוסף, QUIC גם תומך ב0-RTT transmission, מה שמאפשר ללקוחות לשלוח מידע מוצפן כבר בפקטה הראשונה בתנאי שהלקוח תקשר לאחרונה עם השרת.



(Taken from https://commons.wikimedia.org/wiki/File:Tcp-vs-quic-handshake.svg)

מבנה הפקטה בQUIC

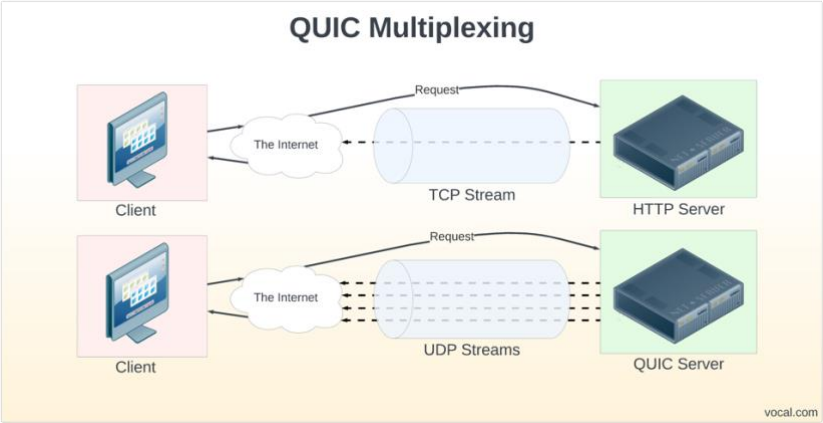
<u>פורמט הפקטה:</u>
ב TCP גודל הheader הוא קבוע. אך בQUIC ישנם שני סוגים שונים של headers. במהלך יצירת החיבור, QUIC משתמש בheader גדול כדי לכלול מידע הכרחי. לאחר יצירת החיבור, QUIC עובר לתבנית header קצרה יותר שמכילה רק את המידע ההכרחי ובכך יעיל יותר. בנוסף, לכל פקטה בQUIC ישנה מזהה ייחודי שמאפשר לשמור על סדר השידור ועוזר בתהליך שחזור הפקטה במקרה והלכה לאיבוד.



(Taken from https://www.f5.com/content/dam/f5-com/nginx-import/primer-QUIC-networking-encryption_stream-anatomy.png)

<u>זרימות:</u>
QUIC משלב ריבוי זרימות אל תוך השכבה התעבורתית בדומה לHTTP/2. בQUIC לכל חיבור יכול להיות מספר זרמים בו-זמנית, המזוהים ע"י stream ids ייחודיים. כל זרם כזה מתנהג בדומה לחיבור TCP כך שזרימת המידע הינה מסודרת. כל מידע בתוך זרם מפוצל לפריימים וכל פריים מזוהה לפי שילוב של הid והoffset שניתן לו - מה שעוזר למנגנוני loss detection של הפרוטוקול לזהות מידע שאבד ולהשלים אותו במידת הצורך.

בנוסף, נשים לב שהשימוש בזרימות מנטרלת את בעיית ה Head-of-Line (HOL) הקיימת בTCP כפי שציינו ב1. במקרה של איבוד פקטות (packet loss) רק הזרם שבו הייתה אבדה יתעכב ואילו שאר הזרימות יזרמו כרגיל.



(Taken from https://vocal.com/wp-content/uploads/2024/02/QUIC-Multiplexing-2048x1066.png)

<u>תמיכה בunreliable datagram delivery:</u>
ישנן אפליקציות, במיוחד אלו שמשתמשות במידע בזמן אמת, המעדיפות לשדר מידע מאובטח באופן לא אמין בתמורה למהירות. אפליקציות מסוג זה, יעדיפו להשתמש בUDP או בDTLS. QUIC גם תומך בשידור מידע מאובטח אך לא אמין באמצעות Datagram Frames. בניגוד לפריימים רגילים, אלו פריימים שלא ישודרו פעם נוספת במקרה וילכו לאיבוד. בהשוואה לUDP או DTLS, QUIC מבצעת לחיצת יד המאפשרת שידור מאובטח אך לא אמין.

מה QUIC עושה כאשר פקטות מגיעות באיחור או לא מגיעות בכלל?

QUIC משתמש בזיהוי אובדן מבוסס ACK בכך שכל פקטה מפוצלת למספר של פריימים וכשפקטה היא ACK, הוא מאשר את קבלת הפקטה כולה וכל הפריימים שלה מתקבלים. אם פקטה לא מקבלת ACK אך פקטה מאוחרת יותר תקבל ACK, הפקטה המוקדמת והפריימים שלה ילכו לאיבוד.
QUIC משתמש ב2 דרכים על מנת לקבוע איבוד פקטות:

1. לפי המספר שלהן (לכל פקטה יש מזהה ייחודי)
2. עבר זמן מסוים ולא התקבל ACK עבור פריים מסוים

הדרכים הללו מאפשרות לסידור מחדש של הפקטות ובכך הפרוטוקול מונע שידור מחדש ללא צורך ממשי בכך.
וכשפקטה הולכת לאיבוד, הפריימים שלה ישודרו מחדש בדומה לTCP - באופן אמין ומסודר.

הCongestion Control של QUIC

QUIC מיישם CC באופן דומה לTCP. כפי שציינו קודם, QUIC ימספר כל פקטה וישתמש בoffset בframes על מנת להבטיח אמינות. בנוגע לCC, QUIC משתמש בCC מבוסס חלון, בדומה לTCP. לQUIC אין מימוש ספציפי של אלגוריתם CC מבוסס חלון, ומשאיר את המימוש בצד הלקוח - מה שהופך את התהליך לגמיש יותר.

כדי למנוע הקטנת חלון לא רצויה, QUIC יצמצם את החלון שלו אך ורק אם יזהה עומס. עומס מזוהה אם שתי פקטות אובדות ואין פקטות ביניהן המקבלות ACK. המרווח זמן בין הפקטות האבודות הללו מחושב בשילוב הRTT הנוכחי כדי לקבוע עומס.

מצד הלקוח, הלקוח ימתן את כמות השליחות שלו ע"י שליחת הפקטות שלו במרווחים המבוססים על הRTT הממוצע ובכך יופחת הסיכוי לעומס.

החלק ה"רטוב" בפרויקט - מימוש ריבוי זרימות

מבוא

החלטנו לממש את חלק 1 — ריבוי זרימות כאשר ממשנו שתי תוכניות פייתון: client.py וserver.py. החלטנו שבשרת יהיו מאוחסנים קבצים רבים בפורמט txt. בגדלים שונים וכאשר לקוח מתחבר לשרת, השרת יעביר את ללקוח את כל הקבצי הטקסט שמאוחסנים בו. למרות שאנחנו לא מממשים ריבוי זרימות כפי שהן בQUIC באופן ישיר, הגישה של השרת לטיפול בהעברות קבצים במקביל מדמה את הרעיון של זרמים ב-QUIC.

צד השרת ומימוש ריבוי הזרימות

השרת יוצר סוקט על פורט מסויים ואז מאזין לחיבורים של לקוחות. עם חיבור של לקוח השרת ישלח לו את כל הקבצים שזמינים להורדה. השרת תומך בעד 10 לקוחות במקביל.

לגבי ריבוי זרימות, נזכיר שאנחנו מנסים לדמות זרימות כפי שמתואר במאמר; כל קובץ שישלח ללקוח מהווה זרם, כאשר עבור כל זרם השרת יקצה thread בודד מה שמדמה את הרעיון של מספר זרימות תחת חיבור אחד כפי שהן ממומשות בQUIC.

השימוש בthreads שונים מייעל את הביצועים בצד השרת ומנטרל את בעיית הHoL אך גם מביא אותנו לבעיה הבאה: הזרמים פועלים במקביל אחד לשני וגורמים למידע להישלח ללקוח בצורה לא מסודרת. נסביר כיצד פתרנו את זה בצד הלקוח בהמשך.

חשוב לדעת שתקשורת לקוח-שרת מוגבלת בדרך כלל על ידי רוחב הפס של הרשת. העברת מספר קבצים במקביל על פני threads שונים אינה מגדילה את רוחב הפס (המחשבים המודרניים יודעים למקסם את רוחב הפס..) ומה שקורה בפועל בפרויקט שלנו זה שהשרת מחלק את רוחב הפס הזמין בין threads הקיימים.

צד הלקוח

צד הלקוח הוא פשוט יחסית, הלקוח מתחבר לשרת בעזרת כתובת ופורט ומבקש את כל הקבצים שמאוחסנים בו.
כפי שציינו מקודם, ישנה בעיה שצצה בעת שימוש במספר threads בצד השרת עבור כל קובץ: המידע שהלקוח יקבל אינו מסודר; ייתכן שהוא יקבל חבילה שעליה יש מידע ששייך לקובץ מס׳ 1 אחרי זה חבילה עם מידע ששייך לקובץ מס׳ 3 ואחרי זה שוב חבילה שקשורה לקובץ מס׳ 1.

כדי לפתור את זה, השרת שולח כל פריים בצורה הבאה:

file_name | frame_id |frame_size| file_data

כאשר החלק הירוק הוא הheader והחלק הסגול הוא הdata.

גודל הheader הוא קבוע בגודל 43 בייטים. הלקוח תחילה ישלוף בדיוק 43 בייטים מהסוקט שלו, ואז ישלוף שוב מידע בגודל frame_size מהסוקט (frame_size יוגרל בין 1000 ו2000 בייטים עבור כל קובץ בצד השרת).
ולבסוף, הלקוח יסדר את המידע שהוא קיבל בצד שלו באמצעות הfile_name והframe_id. גישה זו דומה לאופן שבו QUIC מנהלת את המידע בתוך הזרמים שלה; באמצעות הרכבה נכונה בצד הלקוח בהתבסס על frame_id.

הפקטה הראשונה

(הפקטה נלקחה מההקלטה 3_files_test, מס׳ פקטה 1)

כבר בפקטה הראשונה, ה43 בייטים הראשונים מהווים את הheader (מסומן בצהוב בתמונה למעלה). הclient יקרא תחילה 43 בייטים בדיוק מהסוקט, ואז ישלוף שוב מהסוקט מידע בגודל 1705 וידע לקשר אותו לfile1.txt.
לרוב בהקלטות, עבור כל פקטה, הheader יהיה בתחילת המידע. לפעמים יהיו מקרים שנראה כאילו המידע אכן מפוצל וללא header, לדוגמא:

```
E@@`,NF{U
P(R/QyaDC0Y0gO0hHk6eRh1EzUUlP7VkQ0NjpkSKaGR5n6xzIAqSITAQIrR0HUYq4YgMlzbSTg9obdqEy5IfHmF1fl1DfJgAp8cYfwAA7ejlcZbVJVpOFiYNlg54ZIPU9RnB38Al8fReLbr9r5l0mw0AnY7qhOyluF7xONwnaJtSp6i2SGtKolH79Gi5xfSsOzz
PNnVPb1N4xUlwx0zytG6oLCMssncRrQWlQW0IDGC00AdwJTLB5TWg19S78WikhfZphjZFFthbljIETowPwC31XhxkbHiqH0zs1NouDxwve0DVuItnH37WmfiuTungIOPq7kPqRHWqEvsZFFB1TGG8TEvmQ17UtW86ktcxyVfuaIpgDcmhBQkRwehWSG0yMJp1B
OAyrtianasouTXVb1GD3hbBcIbkZ2DLWdblMa1qES6PXcKaCEKkEbFHxVpKvuyJQVEWZB54aBH2v61I3f2aOlaoGZ1GYARNFHqEphfVSQnQ3FCKAkLao2MVLeThsJyOoL l9eJMMzYWUFKcO8DAMVfK4C48KrvkgDT0FY7nC31qcdpOf9B3nflBYBJUY8oqNN2j
v3tNs2vMyfjdZRQgPp90Qu7aqbQOv8H6pB57cYkELSrnICyxwVcGmabaeYMhWKHLgyuqzpqgY3RGykEM0BvA4q8JCMi9WAqFwrfrUUFWp6jM0yWWwrWh
```


(הפקטה נלקחה מהתקלטה 3_files_test, מס' פקטה 100)

אין לנו שליטה במתי הפקטה תתחוך בין השמשים, בגלל שאנחנו משתמשים בTCP כבסיס.
אך בפועל זה לא משפיע על הלקוח - header כבר נשלח ומיד אחריו יגיע המידע נוסף הזה.

כיצד ריבוי זרימות מתבטא ברשת?


```
P(R/file1.txt
0000080170ZGVoAG9hXPAisnnHbQBB4zv2eV6Lvdknxtx8I7fNQo5PbmqbqEPiXDt2S2uaADEIboGiQSoleNOrMmV6RWSg7CpqPH0TCrqHzcrvCKGzBGAU0aalxQdFAXXnqaXdfdTyI2XQX1fTcV6VFvh7yIztzts7XmXWDwlBUnn8ZRa5PeCkPYjAbYJcrZXzZGbP896YX3R2Yv9CyAgTERVgzlsBZDxXdKrxtnLnKmHagwcgD
P8cIdM1O9YQsZDia7cM8Djwok9ZpoiGXiH2EzTFBYZwoDLHKDIEyYHha4G1EbmVG8ySDulkm5cfE33gyekemyjtOlgtirR65BefhSOuBWJZLHaOYD62RyUpnTfGXOmewhbrDzQOSWk1Y6F3sGB5KGRGiiFEmi87f0wHFXi4qfKNZf4RA5f1ERGQa5QtEJcf6xVsQapf2ECSPbR0dbYeplRlDN6ggWE40hLWXEZoFLsszn2G2AH
5fs5TRefeYehzA3mce9T5jNbMMETjQ9cCJXSNeOvCZmzq3WksrREopJy7np1qMxabQsI87tjUeldyV8MNDXJjD2cwBTdgFKXIwHVpedxOlko6XR0WbhiLAgRQnpwjsL4NQVnDRlUTUlijuTrMGFznlDyk21zhRdErGoGXSrtbHuMjBRV487uSarnU2rCtl2Hz7SrQF2nDTDmwFtVgjRr8roQWBJQhdbBqEoAycBdcd52kKOa37
X8oey6inJ76rapQU9J29nVdLSQfLsZ1ckYMIus1L60AuOdduqQ5BfaAwKQS755wCX6KJQtnZ384DaIfqiCZ8XXKDbSidQAQMCYGk9WBZBAnb5folIYjhpz9zgyZMLPwRkMwBRtEfjhtWbbEA3kbiw3gnx35GrHOH6t7t6bmTT65y5uBJpJj179wR5hPVL2IKA1NXXsK9kYSrqShGn9zAXLzROEChSCGGhGuxoZYnOHhjqzz5TF
2XWQp31naq7Pp3hXqM4hoRnz4YqK4EwrWW6kPoMknoR4Kxwv17D6Q1uCiYUrVAcgJOd1xd2wv3Om53pYpKfssoIu8olTOoJE8UlLjd4PpJP2UjvlB7P9J1ZtjBFAy4gHBto5ZAG8TkEeobTzvkYBVdYwzf1zem5HtfaYP93niMxYEG8YBDTqtvQetnY1QzcsgsYulhXAjDfAELAXy
UnOs0pvagwdCT8ap6l4jLtxQMr4KlJeSmCZ9MFe81JMb9O5uEx5BUpe19Mm9QJzZtgvz79Ub4Q7M5sMRgnCa52LqPfngP6bcdGg7w4nOSDvlVpv6ViuYC9yftijWpl7UJqxsyWKpR74mOsGNZCWtBtGI3LzJWgkqpTq6IYsnsQeuMXJDihBzmIEUvAtGaX2A3wdPKQ9ugHGYJC0HX3GCpqfqaqqC5UgVDxTB8FP6LE1EBnI8t9Y
o4tUID1rKNGSHM43ziwvkLpnq2AbhNvKrUODcxxUEVrj8114sLBFAIrSMwavEuT308UKe4e7LiqltkEloeBbMil40jwGZigTko3m9H26f3bC7coflle9TJP8T2OcJSMy8wOeuhFShrYTfcfI117TBpgZGpGdLHDCQ4PbB4Es3jgvFDCsYcMSrhyz3kLiOgK6LTbbkciiAuePLmmXsS4jB2l4RuFN6NSmntrHKYEsp2rfcvtJx
IS12xfO0syyrtKTfile2.txt
0000087180IRMzgtgqAv8KsWiF3Dz8nXV7orbYTdiXh0Nf7F1NFLAZrNCzHMokOHgiBVUTnqwk8mdbJcA5aojYSRO2uvJE0K02ez4jMfWwFefGIk2vKqOrzw7BXCLxEgo4rau7jz9DucwLuGepuq2veqqUFaoopMgH5g7x0BJ4DlJ6CRtzd5MvyHluFKzgYPwbfhq0aUT7WkyiIF2VWaCxnQXONjzFdhlX6xvSY7sAbvHNzgEf2
BkrNYUPpNesIlVze2OOZk6MFXMPHfd4cmAOSg3dz1ioLQirQBoiyHHDOSbivyO2Wa1lJxszzj6gpUhKWeHnoBjn1HIV0C0MffNVb2Eem9WQmy8JXwbYovxJlAIvsUN7sKVKSIapsIlhEXRW4GYYxB57JQUPyjNwAq22KA7JaQwE5OURLuqlQ8TBassRNEQFFzZUd0mPRFhatzh7kMf9vvSMQIOxXSSnh2nHfZahsZG2cVHLnsZjr
Mo5EUliAi2MEbuh6R6w0gQ04jfrvKBomBoNaGG03m1EN4y43SwTlROunZyBlaH4BGk1JjOHJgN6byHvRjcW8DHTniumelgLBIYUbZZA69e9Mh8dfgX7ZNcPY29yrFShrFHKMiuQZy4mWdpcxTz2D29WcQiazV23gHIa7dE3WSLLrn13REP7bvTHYKntIK7TmiTsQxciJXx6HGG3QAta1UMPGr59Ho77TWshWMGf2APzEGDGDJpHdP
wlQ0lEuQjlPOvHcJ8uykkl4eV5cb06LBYIe2h0OKxNOfX2L7uPepeMVUjpqwWiucTuA7xB7KUFJPuCaCCyU0qy6ia6vPmE2hYwa5XrLa3dvvGaa8GrsVOgYisyW0QJP9ds6CcShLP9SjiPUaYYVhVLUGTc63RYz3I3acFwF8x84MtWPcVRvZsesHJYaZxxIit6AWTM2arGB6osBdpUdNMRLZSWEk0sV4xrcZflWNfsEEnau2M9
EZLSKnylIVlKJjwCV7Jkz2fefZGQovymH7yfBaRDEWRegoBLDPvJH3OvbmTck8Dar5o1vANFYUzCSAt0c32wFiLsBqIOE9moC9byMJcrII4Ygbhvkw3S7PodkRiUl9NjhheEsQJfcVLhmgnpzOkcS16FxFQaAO71U0TzfdiBOJsztqkmW6aq3PpbbEOMuW6HSpPP09WLyNVftIcyCkCGYmNVL8ViBFJUbLM1EtEv39t87eKS
HiTUX3vFykcOnqwjy9ChMRiKbTnKvL2iLSpdggQMzah2I15x6GGYFkJehBNhoysVou9YOhBIaBXQpZFB7GJYx1JmdRuijCSyrojqhTklpuawjpM1L77pXo4jeRJA0WCHDVxxezpwc93GkKXqBKwE7tvdgW9Oah3brKpxeBdelGFnMie6tzsNYqXct5fPdoGpgN6HeJ3DWOLZr07v7VFlk8qPiEYSA2BRbw3IQJCJfzHykCLZ2VT
mUNFaWCBPAUwR9Pjz6lZFN2nMHtKDmtGkrf2y0CUOGyIrhP3fTrkH4Q2tahGg1wy5usypdTTGmHNVKP8az06eXqP3exhY8q4lr0VCU4ZXdn9sXEAdJ0AU0HrLPPx7F0mEkOpafs7guItcOWCv7ZJcn3xYcZ1vAKfdDhKEcMzllh3kj3Y6OURItxeIZbJlxcz7JBOWqBHS4hrRZ6M50PDHK29XABnDkiwL0dl7ooy54GluA99ngo
1Qh51ftcLO6LoCPoxpnhwdMv0Zz2wYGgs4zwAFXvnOWTIWohE0AQCQtERKX3q3o82yoPjMnOUmb1Pzr8ZWHAs22YRQcEj2crvKvP5oCbjqeEIfm7jfile1.txt
0000081170S8AkhQTRrjwOB4crFrpoj1DhUxztpbln8KjR2d4kK95yVWBQ8EELfsdGDQ1bZ4lclkB9usPoLYyYDdnnrqxhy77vpnR8Hr7aO82CFvR6Lq93Cl4yZ3vVJgAp5vdrqj1uZxzApRcUmOxdrzOslDVt1WrjWPU6IIq2uGnHre9KwV7D8r0uCNuBWrwWKCWOnjz50EWJePZB14sefqlYDFwQmJXhn1STO3WkLT9ryZu8z
vTZFxFu8KF4CbRVditj82v2x567spHYxl0iSbzU8vS7axFRsS15uX35bKqOzQPcrHZxRlZZomz9XhjjRZPVXPEAgQV87i1w9VTappWCHIOPtwxW7lnEH57jHUoYQLRC6N703FkGAc5VMtOYGkLTl2rooXahLYfJEd09tpXmvzYoIVLYTVhrCeaS89JRvLHIgM5v5ThC3JHSKTnsErpwp4mx40hRVCSCdhadsLupJyRJeRGyuh8g1
Ih3aGYzpnJL77GJmEm8RGbs0OMIZfyHCf8WuYl0APmeyGoSalhl5TwZaARSYnDTB8et3XaBvEKFG5abSbcH9p8rWRi8hPlUnDrY8OhgQPVRyd1avHEU6lt9YoA6uxPMPnyYdYlqJU2lPvx75jHbkRKuJqorCkx04U7p3wvSdKNVTSs3YgqEcg9UBZ8fAVYMJUjj11Nmzto0wJSMqNE8Esw3M63KNnwIDf4CN2tMxo5Sb9mMO
mI5c6583TAQDUk5jaSfMCWOEQO3QRYYlyx360X1656YQscC4ybkMeqt5WyitBsOECV0WvW03ELv5h9W3cAvoPqAxrE4CVLwP8uIRxza3lod9qpu9r6iteg9V3i0uPm0NSsLXJv7zzw1D3pdTMVQKQKTAzlr1jeSNQ7qql8FwKVWWw3Qtqyhfr3uBhoqGOr0iK3hVJrOE2awqm0Y6vkyLnHkyS1xTO5gDIRuFL2pAz4SeNF89oYF
fxAGrHn8NgPvzs8CK5Q10WIFZkie49s3Tyds4qt4D8V76q16O6SAlqez4DYapkuC8MqeMVGE8K6FsTuMzdoURtNpsGj59cyKq0iGHR7EvsNBjywHaQ13MWuuTA3XcH95Nawcrzt1zqQHZ1dsCI5rsAHYhIInvqjIOMUFSFyaMBi36xp6OOL6ISBbL5Ym3sMb3C2VIzW2tNf2eYKPCiouuFutFyB0yOCWa3bUtAksPKZuriWxHzd
nn1pn4zxjavaSSw8AnsDhSnK9SvpvJOw7zufvDZEqsSNe0d8JCdjlaBHivGq4UMJo7iTtuJsHBaBNNGrViZVQgl3eHt3FjTXvXbEMHOTwGbmgj9uXRlHOslKgICgXPi8EY387a9nD5EDjvBqFzmIeHBhcvpt6eSg0TSsln5Gj8oPk2uhsLephLzoqD8BE2tDxpIB7RNS41Gm2MLHOj324xUfpUBCP3tTCC3IQfvfi6dNIofa
bFpQSv37fKJ0KXEAKq5IKZF88JQojoOuS0vdDT4I3wxsvPf9iu9eNtDJ3qsTfV0ywAsfVcUF8HrELcegVhPT8bA8G4liF2Eyy20e7GtIU1YWbSDwhiBk6laD0QmfK8Z2hZezOa2T52qYnFqfsS7OupFiyXOdQLZO8W6ozLJj3yKUh4QHBsn1JeiNVeFsKWErIt2G8JwSXZJLiFTkIXyaii4BpmzQsz7OAIhkKOq8yardUX6ew45
B89DJ9pZoxeJbDi file1.txt
0000082170SfQSYJmMkMINyBKdoO03ask99jK2nJdL3r8NST98XsVyBEnOPapX337lNiE2eQvvezoDL580pL8OYWOMYgikrTK16fmzaQIu8w863toRWNQzn8zkuVRXQQOAz8X1a55psuAZdRveLu9KFrK0SHREXHapoK3BG74lMR4Gigrfgk8FJiyRLTofnQbK6zZgylgx0MlfhnHFdazg8xSNS4ijOffOJdMcNc8kc7R
dAJALFWN8NDU63BFDFStyQNj5pd2N1BOHYDKRx96QZUD8TD76ftCkBxr1k2XBXWhhscxHcSGV31ckfioqaizZaV1ZTTCnqBBDWIOe9EjsBTDw2AxthHVQMkvVGjpyjvpdO865vb15DWJ6p4TIDwxYRv5FdPpC63wS9DCgjLt8itcnfN4eWXQLvA2bqJ6P9ke4mWnbjrpmQsMosWjqOjKQogz8s8bjId3tnz0EWFgDmqZB7vnP
N6kPVlpZmaNBHnFdWYjTMjjwDgYiLMWT7zYM7fHBfcaqNXOWJLbaLYbKLH1bVswW3FXxgt9sOsfjylCQ2iNuMRaRo8SSq3YVK7CPQBUKTp3at45lM2vrGJetwzVqjPgDiaRfRJ8qKZvKcdN9YjjjppqMXsbeffRdjnXY4XdHBTHEnpUQmcPskWqSYwtCf3JXh8SQoWoHx2HXCKdAtitDKBc6nvRARVfySMITKvV2WLnhhBTGP
Gq4RzkZcqdZg47mZoGEY4YY6jS7BAW6cmsSEePTXJFfryjT609qLsbGU8K4bx2J9ixJmvKTZIIT9OaSHh8ulzDyAxSABiQKCBALf4df39ciV5tUBUCga0h2NGozcPRKzMBI7INNHm5QUy9z28DEtmlUlySk2O8RlAkAFe2SdbTrdhTKZjvvdOaESauYa9O6ktfSUW6tREjeX24tsqJXulqcUWxtSQ7UWBClnRBE9QO4JARqTM46
wMIGwlHbqX9qKrRRIHpwChJuSCurBlqP5EfBcXOWAEQez8BweNvjd3uINEJNTI2wULsMANJbLAB3itKklbNoMuqTd8ZBDp8NlqBJZYCSEdOWlMgPtdKRT9eJZfqILrX9Rn8rebeurl94uK8zpmxvmVUWTFKwdTrmU4hYVygFgc3nqrEWoYRWa60dRIvkP4NaIDMlGZYjK9ecIRKG13cI6A6vpZgfijRPMdAqVrCoAiDxtdaFaDl
LI7tiVvNYrif6a7WY3UIY3uiKdBYSFodYARWZIJAr3DnCqbjYzgpmrlJmwsFCuiJQpS9EPgRLMKhD96HH2ilJqFebNXvB8L5ZV6IlF7hPgajUjSjRt87pEDOCHQRpiLrKg32eKkMmPzc4TsQ4gifrdNgMylt6el3bxhmvGjTb9HlmOpaYzZLRATYFjDgiUfmDutrv7EtjOfaXQBjciWos8RbsYOhwyMwlFgDdAN2VQFkxC2KSwSh
54rot61S4JAhFJ1IXzlqUhfHYU1d20WKCPcnKUp3e3GVEMKQD9LzMgGfB6HSpZWtZg6mYE6IpvpdM2PldAidhGbE6jSAHZlz6eCEnRb8UJFCF0zwoAwXC7RQXIRA9k7gHsb03kIYOPes8AldUCDJMVtNrJqb8oKSziULQqXcOWd3DCxg1e59ZbA9nquzNFdJfKVq7ErlGVKHkjaYv8sSv4gd6UlpABjvojMofCjzjDNFWgd95Cbb
io8zByVAQBOUfIi file1.txt
0000083170S9TWSq5cbs0CHqFUlf5Fb23LpnknIgyZ1jTr3tLjGNsSI8wQyWZ6vO7eqffZNwOINfA7y9YKRNiFMvVralDXkQz0d3XT2vvW5keYs89Oaa8irUBCUaJR7rV2qZG17rOVt9a95jgh0cJ3vI4Prrdkq3bS8v4UbWd7ZQw3y6VC9kJGCy5sleyoNCFjQTuoGbTZxao0aUrdF4JrGpQ7w4a1NIUiBst7MOZA6DjTKJF
YXsM7Q86esIMWLKs8dC3kmAq9IignSd780n4lJPvuclriyqixRMqRFHqtGzGPjcepGyytyWIWuSmvCVxP4HRsTOVTIJ1gJRYJtbOUnchwh2HcuO2M9oE54eBH6vDI3783EDnqQc7jYeTGfwzT8nPEIHkTWFRQirfGs70c7xgSt3ieaM2ZesEEwJ80BX2d0eO2RDHBaiI95DnuØK3mjHN2J4VnazfDWhNtktYSzZRTUu9Xh7Da
5lKFjiaIzIPE5A8yAIejzf9SSQTyQWkKGpweB9cry6rtnXAouEPØND6XpJ37OiMOvSCxfq3QbEjoEurRhqUØbN3INrWvtUØpFnNuV0Q5KQKedSKHIihMHfb3NWGb3cCkmhØW7jKZB1FaqVtkRHHzUØAGbLTØW4CyjdLEB5LTvVuUFFkuiENbA38Dui38hRhBkA9D2JtOJWYxØvozxSsP2WVdX2pOwnrQZRuAUtnQUZelSiGVO2gt
Hxre4f1nTqqNvRSmhSkynyOA9qz59zCLlNIrgJCj80JwHnHyRF8WFZQgZ2octRZQQpXqIAIi9I3EKIQBTmW28Ie5289PprXToVGY8gWMNMD8NiDO7RhuUrOOH6Lo6v36IqlrWwwP1iYhgSQ44wz53qpISSdquEVDVs3EfOLEuwLIiaWq9tf5YhLAWIcotDAU3ueUTD0X7AKSvoGU3gMH11K3H
6UmtHpE2d3YlzZSLseFX0GAEurQgJf2YHDTLZBCwZleZY3yy1PppNEmcyz4gKiDszOlIPgJZAQKLzm8Zc9FON7BHEO0sPtAIzYdEsk2B7XOE3qWM1KsOCwKQjpKZ5kpEIncwXqgtvnjnOsN3fAkGAhYtTWpThdVHhFHodgxkUJSwScJ2wyMMWYcoacHzIKjUztzSPH61GRLuT8lHd1sl61QrZEjVptrSJcFZxJPdsRMBtFdnoEhO
q4T8YqgtGPvBsrb4pk1XhGahLEEKdtt3MEBjRdk1xES8KrujfbsoiSlsqixØoaVNEBdK4P1xSyiTTCn8gSRma4NJCGeoC9KM3ss7AD13kVk1kktJkE0JrBGLb44V2PMFJcr01eHX4LlIEDfGPyv6EDHTDtKSiLZqYZPg7u6GMpzPwwxQ867JykQvaDbFscN73ZVihsxzgHHKepYs76SQ3chIGL74Kb3yaxUbQIjuq7Uydz55hs
f9S35UGpTkhWtJG9pKTI3r7O5NK0KqkE1tqbqH7BSzlzTeUZtvsQmsQ1wA7A3MxYoTloMwLixNhvr4QgWfuvZWjN3OEau7joulZD9uvakAl3QOdZVuqdanSEAsJcvNpY7hvCvJXPNLd3DuiJNbZxEvDsqUMK8NOBPgy8x2UBkmFP9l5tkxMwwJ2e7ET79BnWv2fjo4xQDcp62NSKiYdSFF5TbkQ1cIXiA8GEOs20fvxqnEh9aZg
P8JjfO3bXMK2yp file3.txt
0000095159luT6YoEcWwef5lbNDfl9Q3liqfakh6N2fkZsqtZCV0AGTnxltulfyLwr33lrBqjFTvx2NJzbB38yl542nlHaNa8ZbJKZEBuBimYL63prBmonx1xQfbA9aNnakzXthUqsenq0SC3XAJHu6FAzHO5Ld1Vfxn5LD2sfjwtA0m08Zi6wGcW26I2pYfij2X19FsDpPh072glFZeoz1VP5kiWRcJgReUNUmVJAYVaPgcI
7MrysB3o7TXqIyYPhOJW2ej8gZVCjhYua1qc1CTRzfUGcx2ZaFKQIam0c6W4FGrQ7Q98vYxIeuN8OISkI8nNkVNR2O6oyTbkRiGPFDPvehbym9n3gjw4unInJ2ebA2FXl72mLnk0Dn6L0X03bBy2uR4dfA3HptGIUWiVmOprOltWbGIfGFaBVsVAxyGYnSS7I6WgHBkW0uSRbdKVGwztHDWS4q6bKt8vEB5vsS91XzEbgIU6Lpg
OYs1xhX5APrURTxPSKNgTxkqqRhl9k6gWSJFkohDVgWJKy3cM6xZWpC4EGIuVHLhDYPlt04SwyHaj9SrFVPQx9S9yzHEeikzadLTYPlYphBv0SGypERJNrJcAxOzgvYz9ZSCQHjFyn647VSNw4Mc7xuwNF4aeAkoHiijhiQC9hBSnLhdzrQvWfwO7aPLeTtf88P7DNvh8KHvsxOMCIW3R5umQzi5mGttpZGYo3tsTSm078q2jXfS
vLtlYpKlfumD3qVE6v8LAWQ2seUBfpuNfpr3II245bdjHmn9p8DEAQkOdTKHtJiLBaVOkrKToyKZ27JyBGnxQ2jPl2AYnK5lVGcNaM1xpCfCl8l2f6KiRnkz567Sktyq9LQPHZI6wy0vfhFw38qVsV2L4ZaofXuPSgXEeuShism7tWSsgDs8v9591DFQLHtWYvSMTffhoxuwBKB2YSlvOmsVt4oxgOUdPPYTKeYPmhGIV3dVu8ssi
BSAWpHJWSrwHDwrobg9KSeVUa9TG6jbemrqMoSY5RhcA7IOD4lkXpYD5B80AxDXYnDGGYOlJVeKwdkzUUpIsLUXqAbozilQl7RRufl6BnshGUinCctHvucSxLZyzLO53W4QVHQgRLbTmf7bobO9sexh57Yfytgy4YEMNzmcz8yJBR5EdstgzX6m6wwØPZzeFgBGPeh8dfv57n8r5K7idVwZmafVDmmteo2ySTXq6nGKhpLBI3rd
kJTtLCWuwdjNYwLqkuQgpm9uxXp6nMLzToxjAWY48OmOonevEBc8aPWslFS4ApfQqnvmAbwsxOyBQsxjat4qY2nsjZTNLv6HYbu6LZzYjTFeZfdVsk8OytqOaM0epsEDRXiOWkhaGXZEQ1mhlwDkdKKQa4qxmq8fLJrTxFXS5ZoZtPLXJXCuWYNH67cRPXFOieRVi777Ti8G0v7OIs7tpz4pLtI4e8M2NfsVF0a4RZYyFrerM7j
hG4fE6niLoMpYJDAeMJshlDHTzViEYF8JHavmeoUmsQnI7lWjXZb7dnslwP9N9F1uT4qSKBTZgKqOxGmdpZeabbwuS18Z1S5wmFrXLFsPglVdoDZlXym4ssR2ykvkbpfAGBPifo32DohFX3NXsYMLfile3.txt
0000096159Osy0do5dx7qbIiUcdLSWqA9tbJrkivRQeDtmioaFKLIYks5BgT7Eo76cVURUHLtkwLBEQ7xDazhVOwDFHBKn86xH7COh2tzIrcDsbcxRNnFI4ysqlbKzV3ohRepFuIzLa7i6nA2QNKHyJmgF2JWlZ01nyphjvGcphRzuLwzjPM7qØQxyX8nGQUqkCaoXqx0sZi7PKb1daiTInp0GRAFqtKccdoppKARb2bdOls
AuLmIjlOhligHfncoz17Brzw35yfmq8EaQColnLBkuU5dIgBCJ80CjuEp4czkBYIrFWZhEVdNuSjivQWiFHrZ4FXFRTldQUx3mMzHzLXtIrUCIbJugxsAnY6HinEyp9siqZoxuUNl82lZavlMRxFYKs4l5Et5tbOBW7RRtZxCmISlqXXyfSMTIJSk0xRnQXZeoZGTlad7rMjsLOho7zWd4RjNJiTdXE08vuwUQ1ph1FJhqhuQA2
7IoYXBv3p2paYSUoRlfiRfTL5ffsmbtt5O9ZVHneGVJ0lkMZ9ZLJhiUA3TcYC7Q3DUl4eXsHhrp03JkOgnz2StAgFg40xCQx8B2IlBKguN5MEKd874AUEESu5rfvg29S6dAf2np2HQcSchCQdw4eWq4SsFRckOTrZGs6eqFnhwUKZYaN6BFbjQGwXYuN1BTBaCIZr18q5gNePvY9npDj7wcqrIAwaTxbkF2oUrgCTcObEq2SFZX9Q92Tlfoo
wXZzJJSUcg7IuhJwAbEmYbimgdxkf9DL3vog8MCghhxhP9qFPTsXhc9a1PzduhR2WltcZ1xzTyTRfOQ1LvzgxJNØyi4q4ZgKaubbTrAhDOfRqF1Kn17wthYGMZtUN5ihbOCmAmouRzrvbGuXPCWRVdTML576org3jnoLhWMadjjiDQYw3ncen1RF0zbr0XWdGfkZv6Is2YMurCes7CawdYxo5bbHW5KUIqB5EoveH9iCGWh
```


(הפקטה נלקחה מהתקלטה 3_files_test, מס' פקטה 99)

כאן כבר ניתן לראות את השימוש בריבוי זרימות - על גבי פקטה אחת, נמצא מידע שייך למס' רבצים שונים, באופן לא מסודר. הלקוח יודע איך לסדר את הפריימים האלה אצלו ולבסוף לייצר את הקובץ הדרוש ובכך אנחנו מרוויחים ניצול מקסימלי כל כל פקטה בפועל.

שימוש בתקשורת בזמן 0-RTT

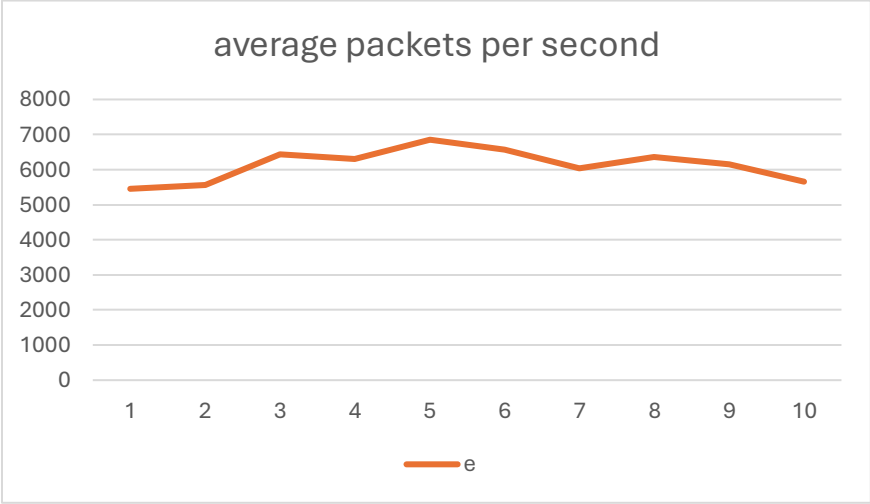
השרת שלנו מחקה תקשורת RTT-0 על ידי שליחת נתונים ביחד כבר בהחילה הראשונה, ללא צורך בלחיצת יד.
המימוש הזה ממזער את הזמן שלוקח להתחיל בהעברת נתונים ודומה מאוד למימוש שקורה בפועל בפרוטוקול QUIC.

ניסויי על מספר זרימות בין 1 ל-10

ראשית, לצורך אחידות הניסוי, יצרנו 10 קבצים זהים בגודל 4mb שאותם נעביר בפרוטוקול שלנו. תחילה, נציג את הנתונים שאספנו:

# of streams (files)	1	2	3	4	5	6	7	8	9	10
d	8294456	8604580	8206837	8633710	8737882	8798714	9015263	9125150	8889883	9528417
e	5454	5560	6430	6298	6853	6573	6042	6363	6155	5648

הגרף של e כפונקציה של מספר זרימות

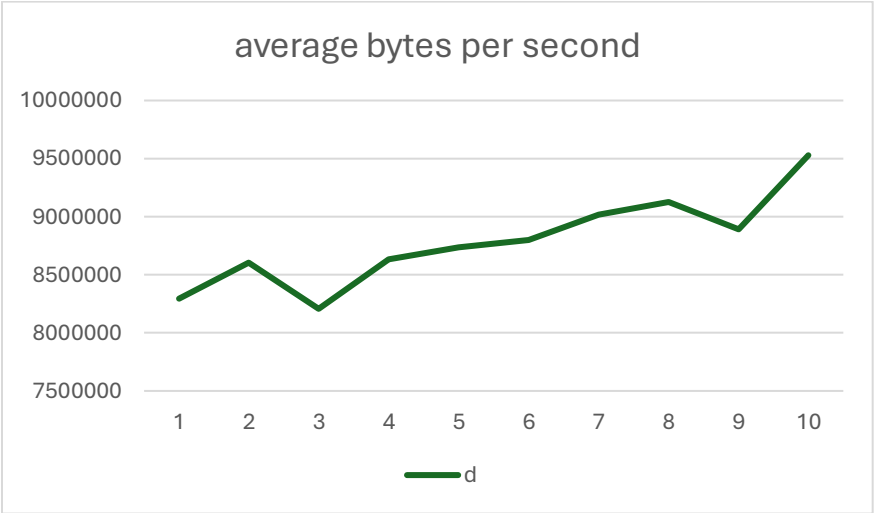


סיכום תוצאות הניסוי - האם יש קשר בין מספר הזרימות למספר הפקטות הממוצע לשנייה?

ניתן לראות תחילה שככול שנוספים זרמים, קצב מספר הפקטות הממוצע לשנייה משתפר. לאחר מכן, כאשר מגיעים למספר זרימות בגודל 5-6 נראה שקצב הרשת מגיע לסוג של מישור ולבסוף, הוספת זרמים נוספים (7-10) גורמות לירידה במספר הפקטות הממוצע לשנייה.

לסיכום, מסתמן שהוספת עד כ-5 זרימות משפרת את מספר הפקטות הממוצע לשנייה יביא אותנו לתוצאות האופטימליות ביותר. לאחר מכן, הוספת מס׳ קטן של זרימות נוספות לא תשפיע אך ככול שנוסיף יותר ויותר זרימות כך ביצועי הרשת ידעכו, ככול הנראה בגלל שהמערכת כבר הגיעה לגבולה ב5~ זרימות וכל זרימה לאחר מכן תגרום לעומס.

הגרף של d כפונקציה של מספר זרימות



סיכום תוצאות הניסוי - האם יש קשר בין מספר הזרימות למספר הבייטים הממוצע לשנייה?

נראה שהיו נפילות ספציפיות כאשר הוספנו זרימה נוספת בהינתן 2 זרימות ו8 זרימות. קשה לקבוע מה הסיבה לנפילות אלו וסביר להניח שמדובר בנפילות מקריות בגלל תנאי רשת, כיוון שלא ניתן להניח איזשהו דפוס. סך הכל, מה שכן ניתן לראות בניסוי המצומצם שלנו זה שככול שמוסיפים יותר זרימות, קצב הבייטים הממוצע לשנייה עולה - כך מסתמן שביצועי הרשת ישתפרו (נוכל להעביר יותר מידע בכל שנייה). נצטרך לעשות ניסוי רחב יותר עם מספר גדול יותר של זרימות על מנת לקבוע מה הוא מספר הזרימות האופטימלי.

לקינוח: אי-יעילות ידועות בפרויקט שלנו

- אי-יעילות בגודל הheader: בכל פקטה, אנו שולחים עימה סוג של header שמציין את שם הקובץ, גודל החבילה שהוגרל ע"י השרת ועוד. חלק מהפרטים אלו הם קבועים ולכן ניתן לפתור את האי-יעילות הזאת בקלות באמצעות שליחת שם הקובץ וגודל החבילה רק בפקטה הראשונה ואילו בפקטות הבאות להשתמש בheader קטן יותר שמציין רק את מזהה הקובץ והמידע בפקטה - כפי שקורה בפועל בפרוטוקול QUIC.

- כמות מרובה של קבצים: כאשר יהיו לנו כמות גדולה של קבצים, השרת יקצה לכל קובץ thread משלו מה שיצא מכלל שליטה באופן מהיר מאוד וכמובן שזה לא אופטימלי.

- שימוש בTCP כפרוטוקול: כמובן שלא התבקשנו לממש QUIC לגמרי אך חשוב לציין שהשימוש בTCP מגביל אותנו מאוד בהמון מובנים וגם בפועל QUIC בנוי מעל UDP.

נספחים

מרבית התשובות שלנו בחלק היבש מתבססות על המאמר "A Quick Look at QUIC"
https://web.cs.ucla.edu/~lixia/papers/UnderstandQUIC.pdf
https://en.wikipedia.org/wiki/QUIC
https://blog.cloudflare.com/the-road-to-quic
/https://www.chromium.org/quic
ChatGPT