מגישים: אלמוג יעקב, טל סומך

מטלה 3 – תקשורת

חלק א'

1. יתרון אחד לשימוש ב-DoH הוא העובדה שה-DoH מוגדר להעביר בברירת המחדל שלו סגמנטים תחת פרוטוקול TCP בשונה משימוש ב-DNS רגיל שמעביר בברירת המחדל שלו סגמנטים תחת פרוטוקול UDP
יתרון זה בולט במקרים בהם ישנם הרבה בעיות ברשת באופן קבוע ובעקבות זאת הרבה סגמנטים של UDP הולכים לאיבוד ומשאבים רבים מנוצלים לריק. זאת בניגוד ל-DoH ששולח את הסגמנטים תחת פרוטוקול TCP ובמקרה של בעיות ברשת הפרוטוקול מנהל את זמני השליחה מחדש בצורה יעילה ומהירה יותר.

2. א. חסרון אחד בשימוש ב-DoH הוא שבעקבות האבטחה וההצפנה שהוא מספק ישנה עקיפה של סינונים שהרשת עושה ועם זאת נמנעת יעילות סינון הרשת הנרכשת מתובנות המסופקות מהגלישה.
ב. חסרון נוסף הוא שינוי בהרגלי הגלישה הבא לידי ביטוי בכך שתכנים רבים מפוזרים ברחבי הרשת באתרים שונים ובזמן שה-DoH יבזבז משאבים אחרי נסיון גישה לאתר ספציפי (תחת פרוטוקול TCP כפי שהסברנו בתשובה 1), הגולש היה יכול להשיג תכנים אלו באתרים אחרים.

3. נציע פתרון לחסרון הראשון שהזכרנו.
על מנת לספק סינון ברשת תוך כדי שימוש בDoH ניתן להציע פתרון ברמת הרשת הפרטית כגון סינון ברמת האפליקציה (אם מימוש ה-DoH מוגדר ברמת האפליקציה) וכן בכל מימוש DoH ניתן להציע מנגנון המבצע סינון ברשת ברמתו ובנוסף יכול להתעדכן על מנת לספק את היעילות הנרכשת מהגלישה הנצברת (כדוגמת האנטי וירוס).

4. <u>יתרונות מימוש DoH ברמת באפליקציה:</u>
   א. מימוש נגיש ומהיר ע"י הגדרות הדפדפן.
   ב. דינאמי וניתן להתאמה עבור אתרים מסויימים.
   <u>חסרונות מימוש DoH ברמת באפליקציה:</u>
   א. חלקי למשתמש וניתן לעקיפה ע"י אפליקציה אחרת (כך שהאבטחה אינה מובטחת ברשתות עסקים).
   ב. אי נוחות ניהול האפליקציה ברשתות עסקים עם מספר רב של מחשבים.

   <u>יתרונות מימוש DoH ברמת שרת פרוסי ברשת:</u>
   א. מימוש נוח עבור מספר גדול של לקוחות (רשתות עסקים) ע"י ניהול רשת מרכזית אחת.
   <u>חסרונות מימוש DoH ברמת שרת פרוסי ברשת:</u>
   א. אין הצפנה ואבטחה עד לרמת השרת.
   ב. הארכת מהירות הגלישה בעקבות רשת מרכזית זאת.

   <u>יתרונות מימוש DoH ברמת שרת פרוסי מקומי:</u>
   א. מימוש נגיש ומהיר.
   ב. חסכון בתוספת רשת מרכזית.
   <u>חסרונות מימוש DoH ברמת שרת פרוסי מקומי:</u>
   א. אי ניצולת הפרטיות במידה והשרת מנוהל ע"י חברה חיצונית.

יתרונות מימוש DoH ברמת פלאגין במחשב:

א. מספק הגנת עקיפה ע"י התקנת הפלאגין וחסימת ההרשאות אליו.

חסרונות מימוש DoH ברמת פלאגין במחשב:

א. יכולת אי ההתאמה של הפלאגין לאפליקציות המחשב.

ב. אי נוחות ניהול הפלאגין ברשתות עסקים עם מספר רב של מחשבים.

השיטה המועדפת מבין הארבעה הינה מימוש DoH ברמת שרת פרוסי מקומי מכיוון שעם היתרונות של נגישות ומהירות המימוש (וכמובן יתרונות הבסיס של ה-DoH) ניתן להמנע מהחסרונות שהוזכרו במימוש זה ע"י ניהול השרת ע"י חברה אמינה או חברת בית.

מקור עזר: https://www.noip.com/blog/2019/10/30/doh-pros-cons-dns-https/

5. פרוטוקול ה-TCP מאפשר ל-DoH לפעול לפעמים בצורה יותר מהירה מאשר 53Do, יתרון זה בולט במקרים בהם ישנם הרבה בעיות ברשת באופן קבוע. לדוגמא, פסק הזמן המוגדר כברירת מחדל לבקשות 53Do בלינוקס מוגדר לחמש שניות שהוא הזמן המוקדם ביותר שאחריו תועבר שאילתת DNS נכשלה. זאת בניגוד ל-DoH המבוסס על פרוטוקול TCP ולכן הסגמנטים עשויים להיות מועברים מחדש באופן אוטומטי לאחר זמן של 2XRTT (לשרת הרקורסיבי). לפיכך, DoH יוכל לשחזר במהירות רבה יותר שאילתות DNS שאבדו מאשר 53Do.

מקור עזר: https://labs.ripe.net/author/austin_hounsel/analysing-the-costs-and-benefits-of-dns-dot-and-doh-for-the-modern-web/

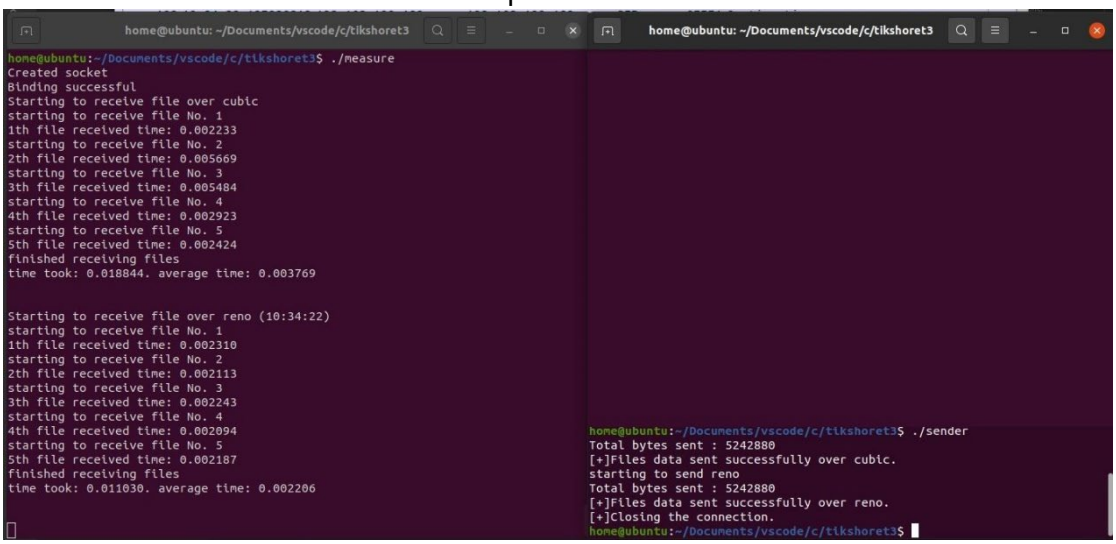
בס"ד

מגישים: אלמוג יעקב, טל סומך

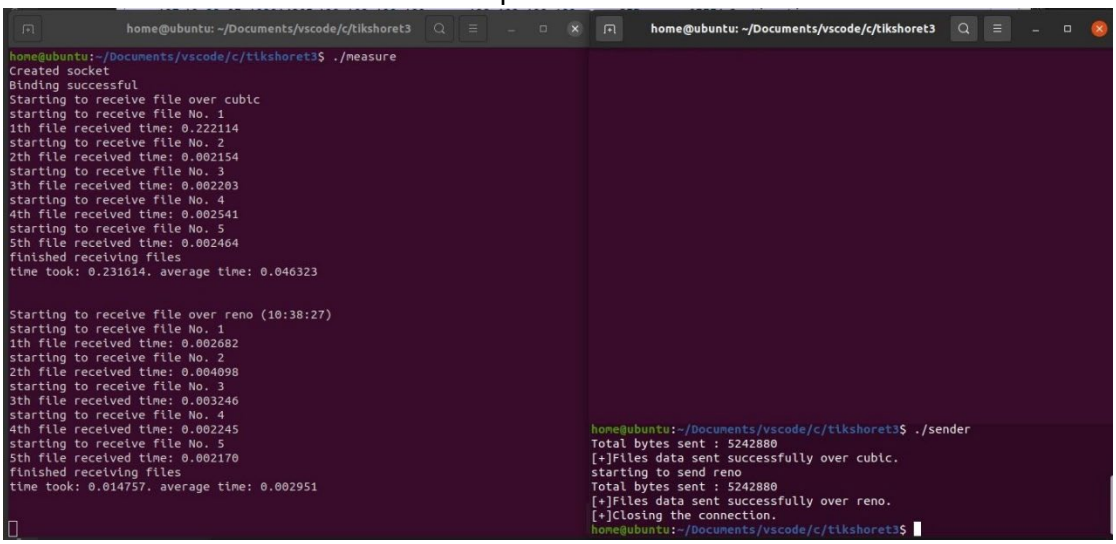
חלק ב'

תצלומי מסך:

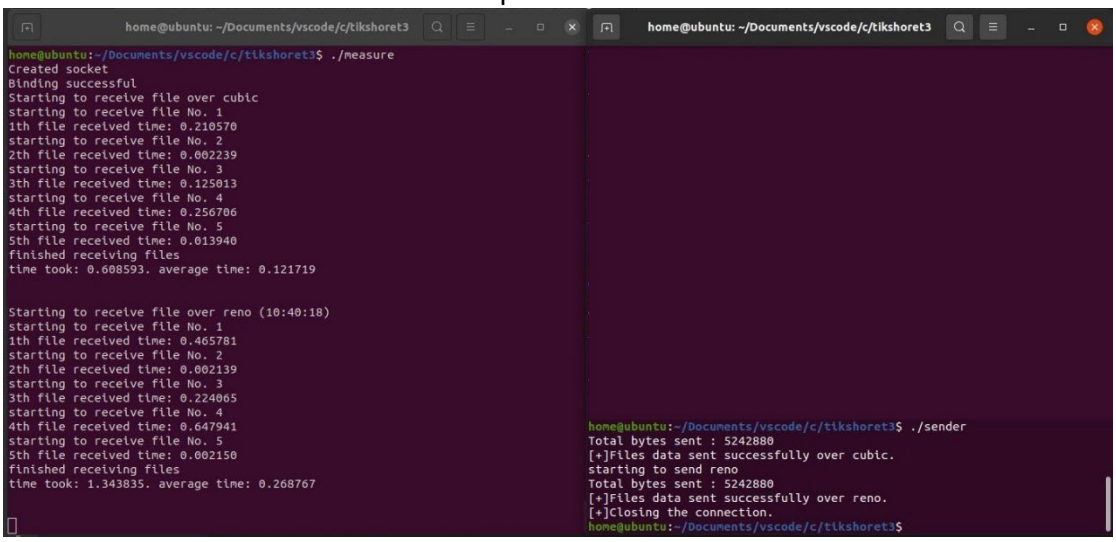
ללא איבוד פקטות:



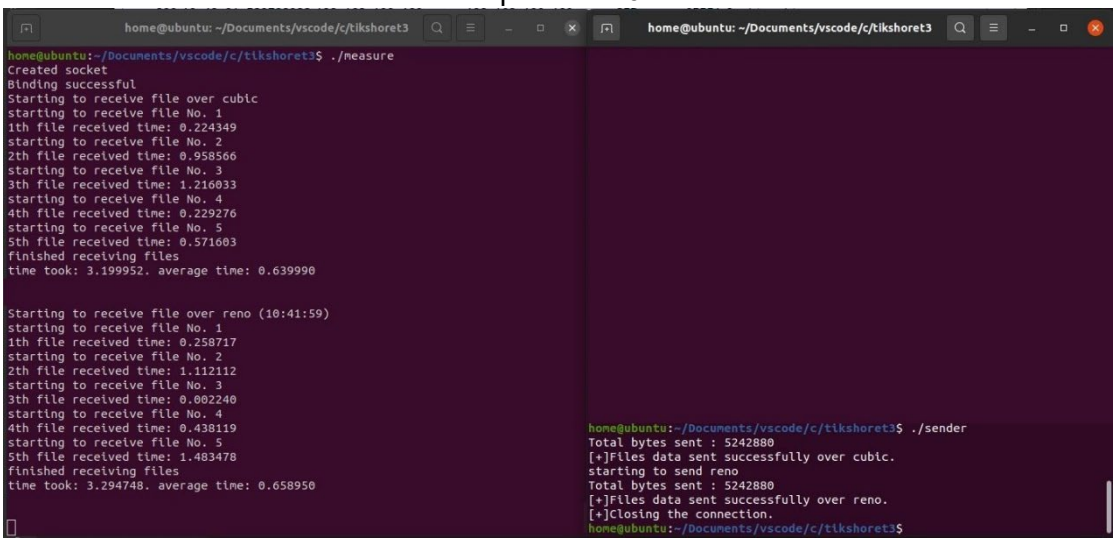
10% איבוד פקטות:



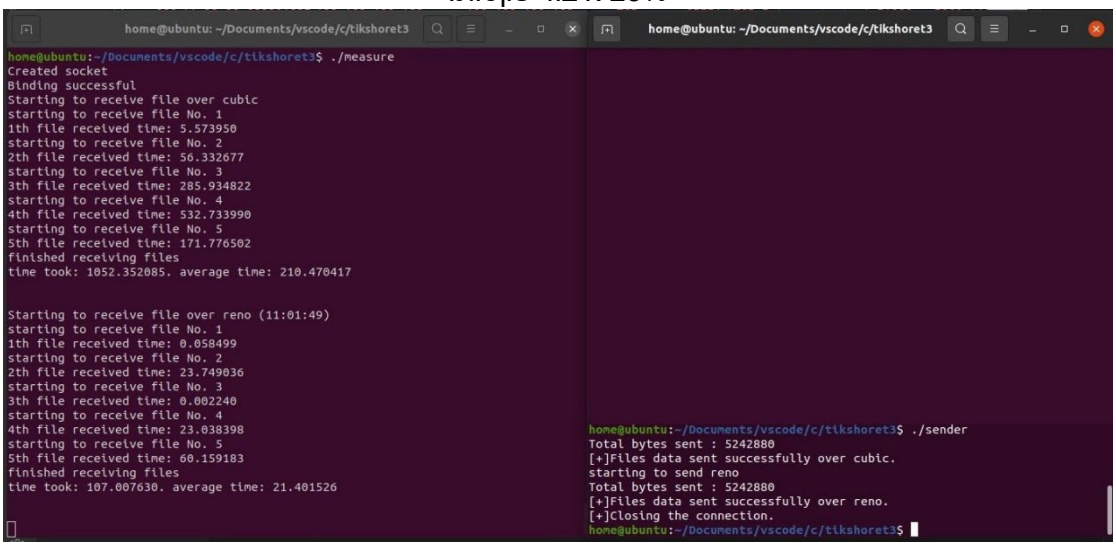
15% איבוד פקטות:



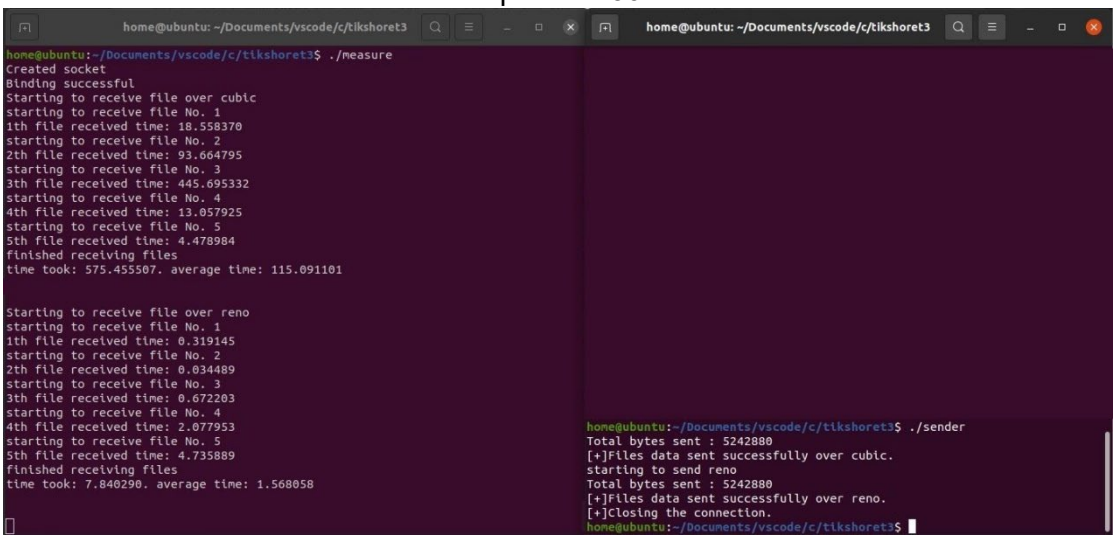
20% איבוד פקטות:



25% איבוד פקטות:



30% איבוד פקטות:



טבלת מדידות:

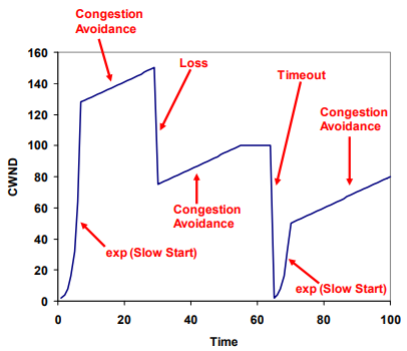
איבוד פקטות באחוזים	זמן הגעה ממוצע (Cubic)	זמן הגעה ממוצע (Reno)
0	0.003769	0.002206
10	0.046323	0.002951
15	0.121719	0.268767
20	0.639990	0.658950
25	210.470417	21.401526
30	115.091101	1.568058

נזכיר כי פונקציית איבוד הפקטות הינה רנדומלית ולכן אין עקביות בזמנים הממוצעים כפי שניתן לראות בתמונות הנ"ל (כדוגמת אחוז איבוד הפקטות תחת אלגוריתם Reno ב-30% איבוד).

ניתוח נתונים

הגדרות עזר:

Symbol	Definition
cwnd	Current congestion window in number of packets
$cwnd_{INT}$	Initial congestion window (we use $cwnd_{INT}$ = 2 packets)
sst	Current slow-start threshold in number of packets
sst_{MAX}	Threshold (in packets) to switch from exponential to logarithmic increase (varies with experiment)
sst_{INT}	Threshold (in packets) to terminate initial slow start (varies with experiment)



TCP Retransmission: זהו מנגון השליחה מחדש שעובד כאשר אנחנו לא מקבלים Ack על TCP Segments בזמן המוגדר RTO ולכן הוא משדר מחדש.

Fast Retransmission: במצב שבו יש מספר רב של Ack ואז נוצר מצב של Dup Ack אז המחשב שולח חזרה את ה-Segment הדרוש ב-Fast retransmission

RTO: מכל ה-Retransmission נוצרת לנו בעיה שזמן ה-RTO ממשיך לעלות, זהו מנגנון שנוצר על מנת למנוע עומס.

מכיוון שהפרוטוקול עובד בכך שהוא אוגר זמן לפני שהוא שולח שוב, ניתן לראות שבשליחה הראשונה הוא יעמוד על זמן קצר של שניה בעוד שכבר בשליחה העשירית ה-RTO יכול להגיע לזמן של כ-30 שניות ואף יותר.

TCP Previous Segment Not Captured: זה שגיאה של Wireshark שהוא לא מזהה פאקטה שמתאימה ל Segment שציפה, מכיוון שביצענו באופן מלאכותי איבוד פקטות אז שגיאה כזו מובנית.

Dup Ack: מנגון זה הוא חלק ממנגנון תיקון השגיאות של TCP ששולח את הDUP ACK מתי שהוא מזהה פקטה שלא מגיעה בסדר שמצופה שתגיע.

מסקנות:
ניתן לראות מהתמונות הנ"ל (המתעדות את שליחת וקבלת הקבצים בהתאם לאחוזי איבוד הפקטות) כי מיטבות השליחה גדלה עם כל שליחה מחדש תוך כדי זיהוי מגבלות הרשת והתאמה למגבלות אלו. השליחה בתחילה מתבצעת בחלונות גדולים ובתדירות גבוהה ואז גודל החלון מצטמצם ותדירות השליחה קטנה בהתאם לאיבוד הפקטות.
משתנה זה נקרא Congestion window (RWND) שהוא משתנה של מצב TCP המגביל את כמות הנתונים שה- TCP יכול לשלוח לרשת לפני קבלת ACK. חלון זה (RWND) הוא משתנה המפרסם את כמות הנתונים שצד היעד יכול לקבל.
<u>ההבדלים בין רינו לקיוביק:</u>

שני האלגוריתמים דומים בשלבי ה-Fast recovery, Fast retransmit וה-Slow start.
השוני בין השניים הוא השינוי בחלון העומס שלהם.
באלגוריתם Reno חלון העומס משתנה בצורה לינארית, כאשר הוא מקבל 3*Dup הוא מוריד בצורה יחסית את חלון העומס לרמת
ה-Ssthresh (שזהו בעצם Threshold שמשתנה לפי מצב איבוד הפקטות שלנו) ומתחיל לחזור באופן לינארי.
לעומת זאת באלגוריתם Cubic חלון העומס משתנה לפי פונקציה של זמן מהעומס האחרון שלנו:

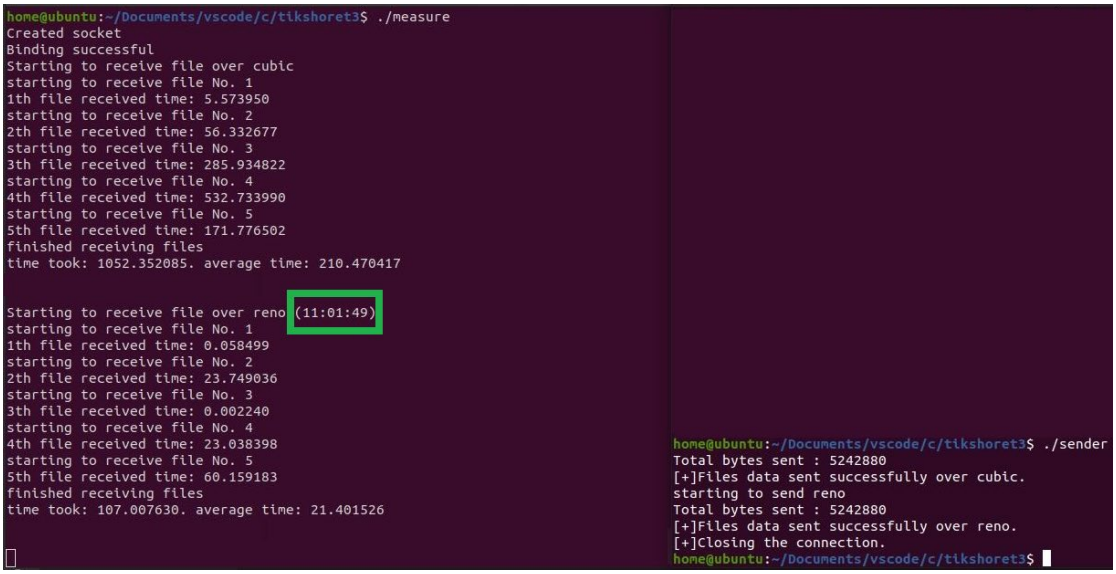
$$W(w_0, t) = C(t - K)^3 + w_0$$

כאשר הערכים K, w_0 משתנים לפי הורדת חלון העומס האחרון שלנו והערך C הוא קבוע.

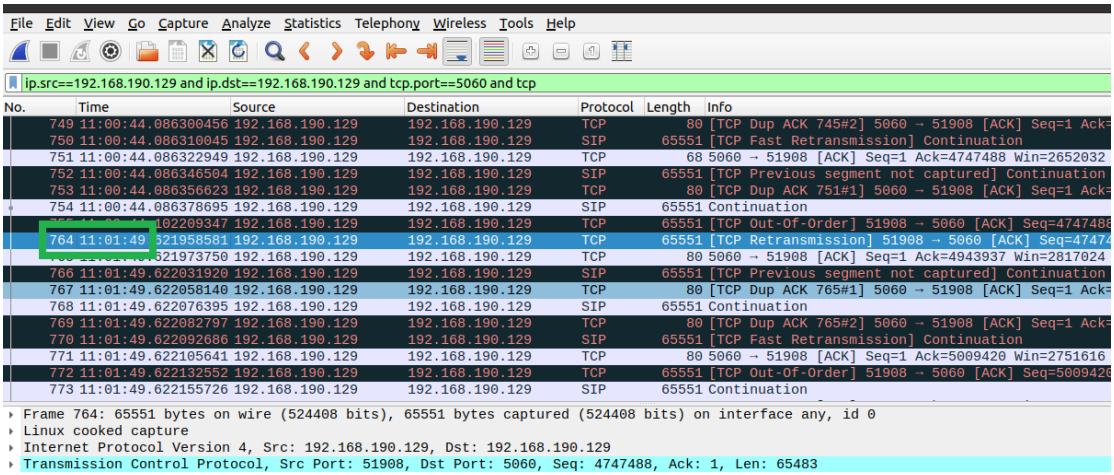
שני האלגוריתמים דומים גם בכך ששניהם ממשים את האלגוריתם של AIMD שמשנה את חלון העומס במצב של timeout או שהמקור קיבל 3*Dup-Ack.

<u>כעת עלינו להראות בעזרת ויירשארק את השוני בין שני האלגוריתמים</u>
<u>(באמצעות גרף סטטיסטי כפי שנאמר בשיעור)</u>
נראה את שוני זה בתרחיש של 25% איבוד פקטות.

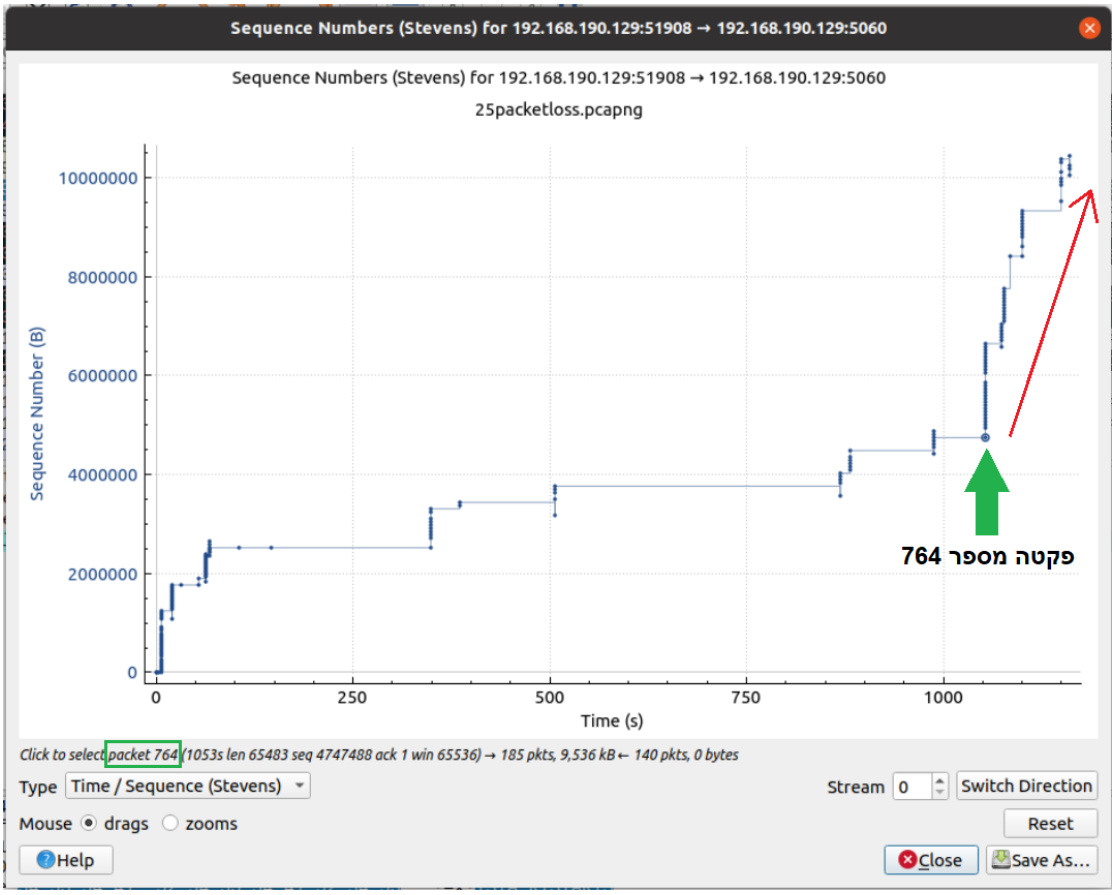
נשים לב כי בצד ה-Measure שינוי האלגוריתם ל-Reno התבצע בשנייה 11:01:49



לכן בהכרח החל מפקטה מספר 764 קבלת הפקטות מתבצעת תחת אלגוריתם Reno:



כעת ניתן להבחין בעליה הלינארית תחת אלגוריתם Reno:



העלייה הלינארית מושפעת מעליית מספר ה-Sequences ביחס לזמן.
סך הפקטות הכללי הינו 916 כפי שניתן לראות בקובץ ה-Pcap המצורף ולכן מספר הפקטות בהם השליחה והקבלה היו תחת אלגוריתם Reno קטן ביחס לסך הפקטות הכללי ובכל זאת ישנה עליה גדולה לינארית.
מה שמעיד על חלון עומס גדול ביחס לאלגוריתם Cubic המושפע (ע"י פונקציה) מהעומס הקודם (וכמובן שהחלון יהיה קטן יותר מכיוון שאנו תחת 25% איבוד פקטות).