

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 מנעת יעילות סינון הרשת הנרכשת מתבונות 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 ב-DoH 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. יתרונות מימוש DoH ברמת באפליקציה:
 א. מימוש נגיש ומהיר ע"י הגדרות הדפדפן.
 ב. דינאמי וניתן להתאמה עבור אתרים מסויימים.
חסרונות מימוש DoH ברמת באפליקציה:
 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.
- יתרונות מימוש DoH ברמת שרת פרוסי ברשת:
 א. מימוש נוח עבור מספר גדול של לקוחות (רשתות עסקים) ע"י ניהול רשת מרכזית אחת.
חסרונות מימוש DoH ברמת שרת פרוסי ברשת:
 א. אין הצפנה ואבטחה עד לרמת השרת.
 ב. הארכת מהירות הגלישה בעקבות רשת מרכזית זאת.
- יתרונות מימוש DoH ברמת שרת פרוסי מקומי:
 א. מימוש נגיש ומהיר.
 ב. חסכון בתוספת רשת מרכזית.
חסרונות מימוש DoH ברמת שרת פרוסי מקומי:
 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

חלק ב'

תצלומי מסך:

ללא איבוד פקטות:

```
home@ubuntu: ~/Documents/vscode/c/tikshoret3$ ./measure
Created socket
Binding successful
Starting to receive file over cubic
starting to receive file No. 1
1th file received time: 0.002233
starting to receive file No. 2
2th file received time: 0.005669
starting to receive file No. 3
3th file received time: 0.005484
starting to receive file No. 4
4th file received time: 0.002923
starting to receive file No. 5
5th file received time: 0.002424
finished receiving files
time took: 0.018844. average time: 0.003769

Starting to receive file over reno (10:34:22)
starting to receive file No. 1
1th file received time: 0.002310
starting to receive file No. 2
2th file received time: 0.002113
starting to receive file No. 3
3th file received time: 0.002243
starting to receive file No. 4
4th file received time: 0.002094
starting to receive file No. 5
5th file received time: 0.002187
finished receiving files
time took: 0.011030. average time: 0.002206

home@ubuntu:~/Documents/vscode/c/tikshoret3$ ./sender
Total bytes sent : 5242880
[+]Files data sent successfully over cubic.
starting to send reno
Total bytes sent : 5242880
[+]Files data sent successfully over reno.
[+]Closing the connection.
home@ubuntu:~/Documents/vscode/c/tikshoret3$
```

10% איבוד פקטות:

```
home@ubuntu:~/Documents/vscode/c/tikshoret3$ ./measure
Created socket
Binding successful
Starting to receive file over cubic
starting to receive file No. 1
1th file received time: 0.222114
starting to receive file No. 2
2th file received time: 0.002154
starting to receive file No. 3
3th file received time: 0.002203
starting to receive file No. 4
4th file received time: 0.002541
starting to receive file No. 5
5th file received time: 0.002464
finished receiving files
time took: 0.231014. average time: 0.046323

Starting to receive file over reno (10:38:27)
starting to receive file No. 1
1th file received time: 0.002682
starting to receive file No. 2
2th file received time: 0.004098
starting to receive file No. 3
3th file received time: 0.003246
starting to receive file No. 4
4th file received time: 0.002245
starting to receive file No. 5
5th file received time: 0.002170
finished receiving files
time took: 0.014757. average time: 0.002951

home@ubuntu:~/Documents/vscode/c/tikshoret3$ ./sender
Total bytes sent : 5242880
[+]Files data sent successfully over cubic.
starting to send reno
Total bytes sent : 5242880
[+]Files data sent successfully over reno.
[+]Closing the connection.
home@ubuntu:~/Documents/vscode/c/tikshoret3$
```

15% איבוד פקטות:

```
home@ubuntu:~/Documents/vscode/c/tikshoret3$ ./measure
Created socket
Binding successful
Starting to receive file over cubic
starting to receive file No. 1
1th file received time: 0.210570
starting to receive file No. 2
2th file received time: 0.002239
starting to receive file No. 3
3th file received time: 0.125013
starting to receive file No. 4
4th file received time: 0.256706
starting to receive file No. 5
5th file received time: 0.013940
finished receiving files
time took: 0.608593. average time: 0.121719

Starting to receive file over reno (10:40:18)
starting to receive file No. 1
1th file received time: 0.465781
starting to receive file No. 2
2th file received time: 0.002139
starting to receive file No. 3
3th file received time: 0.224065
starting to receive file No. 4
4th file received time: 0.647941
starting to receive file No. 5
5th file received time: 0.002150
finished receiving files
time took: 1.343835. average time: 0.268767

home@ubuntu:~/Documents/vscode/c/tikshoret3$ ./sender
Total bytes sent : 5242880
[+]Files data sent successfully over cubic.
starting to send reno
Total bytes sent : 5242880
[+]Files data sent successfully over reno.
[+]Closing the connection.
home@ubuntu:~/Documents/vscode/c/tikshoret3$
```

20% איבוד פקטות:

```
home@ubuntu: ~/Documents/vscode/c/tikshoret3$ ./measure
Created socket
Binding successful
Starting to receive file over cubic
starting to receive file No. 1
1th file received time: 0.224349
starting to receive file No. 2
2th file received time: 0.958566
starting to receive file No. 3
3th file received time: 1.216033
starting to receive file No. 4
4th file received time: 0.229276
starting to receive file No. 5
5th file received time: 0.571603
finished receiving files
time took: 3.199952. average time: 0.639990

Starting to receive file over reno (10:41:59)
starting to receive file No. 1
1th file received time: 0.258717
starting to receive file No. 2
2th file received time: 1.112112
starting to receive file No. 3
3th file received time: 0.002240
starting to receive file No. 4
4th file received time: 0.430119
starting to receive file No. 5
5th file received time: 1.483478
finished receiving files
time took: 3.294748. average time: 0.658950

home@ubuntu:~/Documents/vscode/c/tikshoret3$ ./sender
Total bytes sent : 5242880
[+]Files data sent successfully over cubic.
starting to send reno
Total bytes sent : 5242880
[+]Files data sent successfully over reno.
[+]Closing the connection.
home@ubuntu:~/Documents/vscode/c/tikshoret3$
```

25% איבוד פקטות:

```
home@ubuntu: ~/Documents/vscode/c/tikshoret3$ ./measure
Created socket
Binding successful
Starting to receive file over cubic
starting to receive file No. 1
1th file received time: 5.573950
starting to receive file No. 2
2th file received time: 56.332677
starting to receive file No. 3
3th file received time: 285.934822
starting to receive file No. 4
4th file received time: 532.733990
starting to receive file No. 5
5th file received time: 171.776502
finished receiving files
time took: 1052.352085. average time: 210.470417

Starting to receive file over reno (11:01:49)
starting to receive file No. 1
1th file received time: 0.058499
starting to receive file No. 2
2th file received time: 23.749036
starting to receive file No. 3
3th file received time: 0.002240
starting to receive file No. 4
4th file received time: 23.038398
starting to receive file No. 5
5th file received time: 60.159183
finished receiving files
time took: 107.007630. average time: 21.401526

home@ubuntu:~/Documents/vscode/c/tikshoret3$ ./sender
Total bytes sent : 5242880
[+]Files data sent successfully over cubic.
starting to send reno
Total bytes sent : 5242880
[+]Files data sent successfully over reno.
[+]Closing the connection.
home@ubuntu:~/Documents/vscode/c/tikshoret3$
```

30% איבוד פקטות:

```
home@ubuntu: ~/Documents/vscode/c/tikshoret3$ ./measure
Created socket
Binding successful
Starting to receive file over cubic
starting to receive file No. 1
1th file received time: 18.558370
starting to receive file No. 2
2th file received time: 93.664795
starting to receive file No. 3
3th file received time: 445.695332
starting to receive file No. 4
4th file received time: 13.057925
starting to receive file No. 5
5th file received time: 4.478984
finished receiving files
time took: 575.455507. average time: 115.091101

Starting to receive file over reno
starting to receive file No. 1
1th file received time: 0.319145
starting to receive file No. 2
2th file received time: 0.034489
starting to receive file No. 3
3th file received time: 0.672203
starting to receive file No. 4
4th file received time: 2.077953
starting to receive file No. 5
5th file received time: 4.735889
finished receiving files
time took: 7.840290. average time: 1.568058

home@ubuntu:~/Documents/vscode/c/tikshoret3$ ./sender
Total bytes sent : 5242880
[+]Files data sent successfully over cubic.
starting to send reno
Total bytes sent : 5242880
[+]Files data sent successfully over reno.
[+]Closing the connection.
home@ubuntu:~/Documents/vscode/c/tikshoret3$
```

טבלת מדידות:

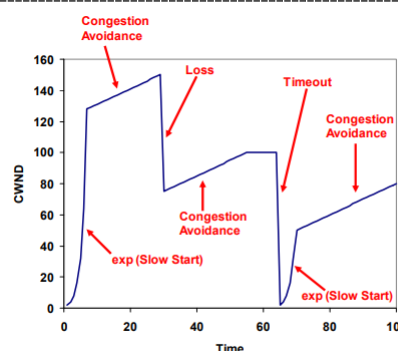
איבוד פקטות באחוזים	זמן הגעה ממוצע (Cubic)	זמן הגעה ממוצע (Reno)
0	0.003769	0.002206
10	0.046323	0.002951
15	0.121719	0.268767
20	0.639990	0.658950
25	210.470417	21.401526
30	115.091101	1.568058

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 מנגנון 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: מנגנון זה הוא חלק ממנגנון תיקון השגיאות של TCP ששולח את ה-DUP 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.

ההבדלים בין רינו לקיוביק:

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 $3 \times \text{Dup}$ הוא מוריד בצורה יחסית את חלון העומס לרמת

ה-Ssthresh (שהוא 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 $3 \times \text{Dup-Ack}$.

קעת עלינו להראות בעזרת וירשארק את השוני בין שני האלגוריתמים (באמצעות גרף סטטיסטי כפי שנאמר בשיעור)
נראה את שוני זה בתרחיש של 25% איבוד פקטות.

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

```
home@ubuntu:~/Documents/vscode/c/tikshoret3$ ./measure
Created socket
Binding successful
Starting to receive file over cubic
starting to receive file No. 1
1th file received time: 5.573950
starting to receive file No. 2
2th file received time: 56.332677
starting to receive file No. 3
3th file received time: 285.934822
starting to receive file No. 4
4th file received time: 532.733990
starting to receive file No. 5
5th file received time: 171.776502
finished receiving files
time took: 1052.352085, average time: 210.470417

Starting to receive file over reno (11:01:49)
starting to receive file No. 1
1th file received time: 0.058499
starting to receive file No. 2
2th file received time: 23.749036
starting to receive file No. 3
3th file received time: 0.002240
starting to receive file No. 4
4th file received time: 23.038398
starting to receive file No. 5
5th file received time: 60.159183
finished receiving files
time took: 107.007630, average time: 21.401526

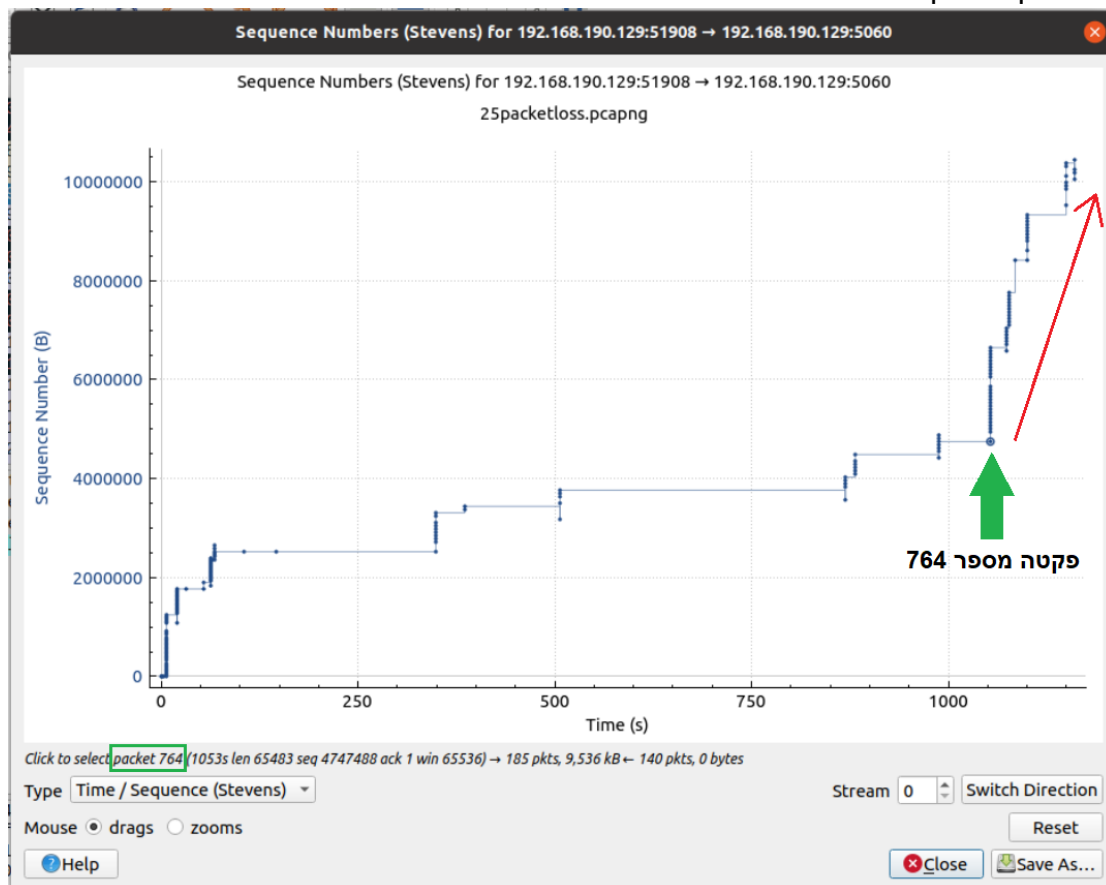
home@ubuntu:~/Documents/vscode/c/tikshoret3$ ./sender
Total bytes sent : 5242880
[+]Files data sent successfully over cubic.
starting to send reno
Total bytes sent : 5242880
[+]Files data sent successfully over reno.
[+]Closing the connection.
home@ubuntu:~/Documents/vscode/c/tikshoret3$
```


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

No.	Time	Source	Destination	Protocol	Length	Info
749	11:00:44.086300456	192.168.190.129	192.168.190.129	TCP	80	[TCP Dup ACK 745#2] 5060 → 51908 [ACK] Seq=1 Ack=
750	11:00:44.086310045	192.168.190.129	192.168.190.129	SIP	65551	[TCP Fast Retransmission] Continuation
751	11:00:44.086322949	192.168.190.129	192.168.190.129	TCP	68	5060 → 51908 [ACK] Seq=1 Ack=4747488 Win=2652032
752	11:00:44.086346504	192.168.190.129	192.168.190.129	SIP	65551	[TCP Previous segment not captured] Continuation
753	11:00:44.086356623	192.168.190.129	192.168.190.129	TCP	80	[TCP Dup ACK 751#1] 5060 → 51908 [ACK] Seq=1 Ack=
754	11:00:44.086378695	192.168.190.129	192.168.190.129	SIP	65551	Continuation
755	11:00:44.086393347	192.168.190.129	192.168.190.129	TCP	65551	[TCP Out-Of-Order] 51908 → 5060 [ACK] Seq=4747488
764	11:01:49.621958581	192.168.190.129	192.168.190.129	TCP	65551	[TCP Retransmission] 51908 → 5060 [ACK] Seq=4747488
765	11:01:49.621973750	192.168.190.129	192.168.190.129	TCP	80	5060 → 51908 [ACK] Seq=1 Ack=4943937 Win=2817024
766	11:01:49.622031920	192.168.190.129	192.168.190.129	SIP	65551	[TCP Previous segment not captured] Continuation
767	11:01:49.622058140	192.168.190.129	192.168.190.129	TCP	80	[TCP Dup ACK 765#1] 5060 → 51908 [ACK] Seq=1 Ack=
768	11:01:49.622076395	192.168.190.129	192.168.190.129	SIP	65551	Continuation
769	11:01:49.622082797	192.168.190.129	192.168.190.129	TCP	80	[TCP Dup ACK 765#2] 5060 → 51908 [ACK] Seq=1 Ack=
770	11:01:49.622092686	192.168.190.129	192.168.190.129	SIP	65551	[TCP Fast Retransmission] Continuation
771	11:01:49.622105641	192.168.190.129	192.168.190.129	TCP	80	5060 → 51908 [ACK] Seq=1 Ack=5009420 Win=2751616
772	11:01:49.622132552	192.168.190.129	192.168.190.129	TCP	65551	[TCP Out-Of-Order] 51908 → 5060 [ACK] Seq=5009420
773	11:01:49.622155726	192.168.190.129	192.168.190.129	SIP	65551	Continuation

Frame 764: 65551 bytes on wire (524408 bits), 65551 bytes captured (524408 bits) on interface any, id 0
 Linux cooked capture
 Internet Protocol Version 4, Src: 192.168.190.129, Dst: 192.168.190.129
 Transmission Control Protocol, Src Port: 51908, Dst Port: 5060, Seq: 4747488, Ack: 1, Len: 65483

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).