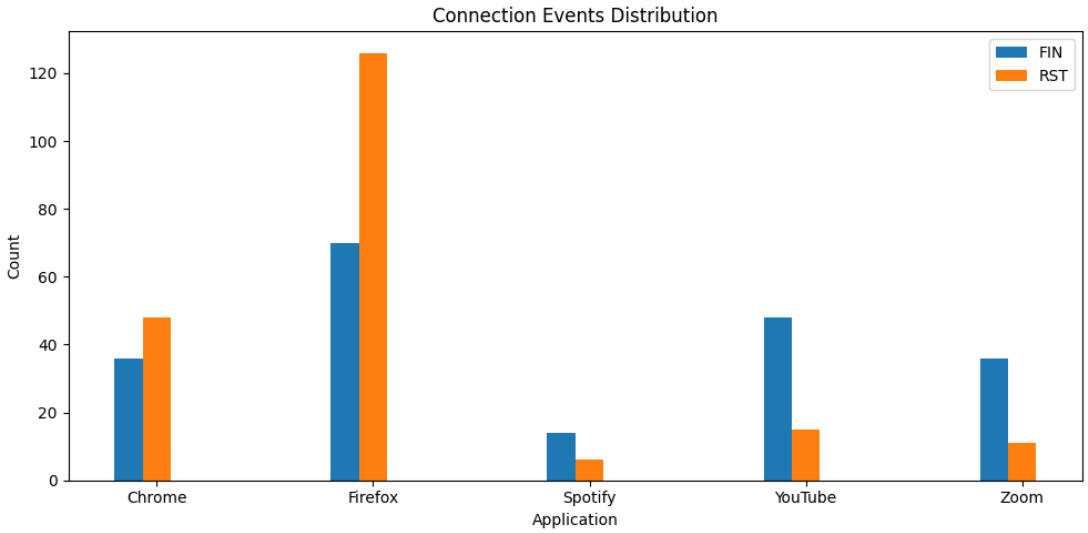
"Connection Events Distribution"



הגרף שלעיל מציג את התפלגות אירועי סיום של חיבור (TCP Termination Events) עבור חמש אפליקציות נפוצות: Chrome, Firefox, Spotify, YouTube ו- Zoom. בפרט, הוא מתמקד בשני סוגי אירועים:

- **FIN Flag** – מסמן סיום תקני ומסודר של חיבור TCP, בו הצד היוזם שולח בקשה לסגירה מתואמת.
- **RST Flag** – מסמן סיום מידי ולא תקני של החיבור (Reset), בדרך כלל מסיבה של שגיאה או סיום כפוי.

אפליקציה	התנהגות	**ניתוח**
Firefox	כמות גבוהה מאוד של RST	ייתכן שנובע מהתנהלות אגרסיבית בפתיחה/סגירה של משאבים, שמפסיקים חיבורים בפתאומיות (למשל, טעינה מקבילית של עשרות משאבים בדף אינטרנט).
Chrome	יחס גבוה של RST לעומת FIN	התנהגות דומה ל Firefox, אך מתונה יותר. עדיין ניתן להבחין באיפוס חיבורים רבים, סממן לסיום לא תקני

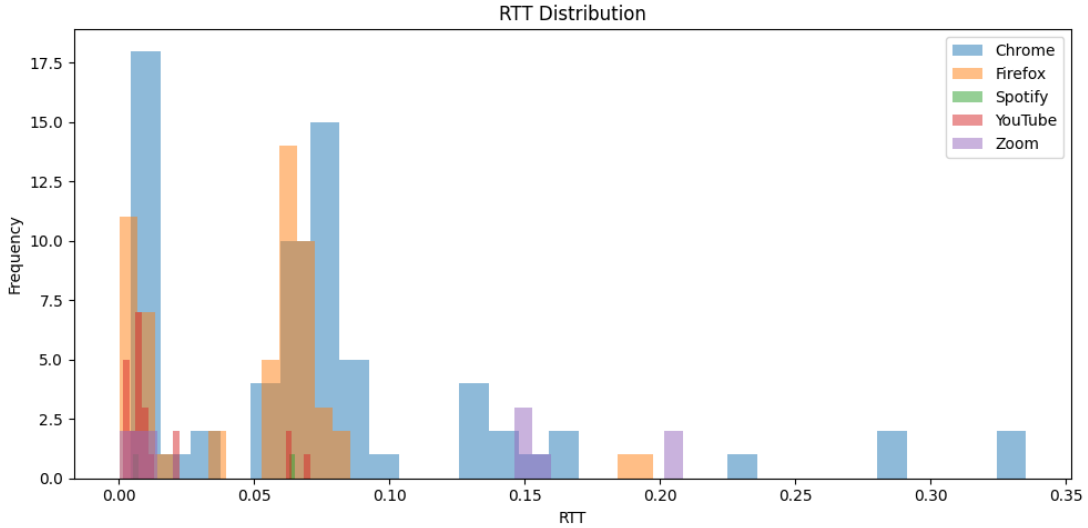
Spotify	בעיקר ,FIN מעט מאוד RST	מעיד על שימוש בחיבורים מתמשכים ויציבים לצורך סטרימינג.
YouTube	כמות גבוהה של ,FIN לצד RST מועט	סטרימינג וידאו לרוב מתבצע על גבי חיבורים יציבים, אך יתכנו חיבורים נלווים לפרסומות דבר המעיד על הופעה של RST
Zoom	כמעט כל הסיומים הם FIN	מצביע על שמירה על חיבור רציף ותקשורת תקינה,קריטי לניהול שיחה בזמן אמת.

- **מה הגרף מציג ?**כמות האירועים (Events) שונים שנמצאו בכל אפליקציה – למשל FIN־ו־RST.
- **מה ההבדל בין האפליקציות?** גרף זה מציג את מספר האירועים (כגון ,FIN, RST, SYN) לכל אפליקציה. אפליקציה שחותכת את החיבורים בצורה מסודרת תציג יותר FIN (סגירה מסודרת), בעוד שאפליקציה עם בעיות חיבור או סגירה לא תקינה תציג יותר RST.
- **למה זה קורה?** סגנון סגירת החיבורים תלוי במאפייני האפליקציה – למשל, דפדפנים שמסיימים חיבורים בצורה מסודרת לעיתים קרובות יראו יותר FIN, ואילו שירותים שמנסים לשמור על חיבור פעיל באופן רציף או שיש להם בעיות רשת עשויים להציג יותר RST (איפוס חיבור).
- **מה אפשר ללמוד מהגרף?**
  - האם האפליקציה סוגרת חיבורים בצורה נקייה (FIN) או שנגרמות הרבה סגירות מיידיות.(RST)
  - כמות RST גבוהה יכולה לרמוז על סגירה כפויה או על בעיות תקשורת
  - אפליקציות תקשורת בזמן אמת (Zoom) ואפליקציות בידור (,Spotify YouTube) שואפות לשמר חיבורים פתוחים לאורך זמן, ולכן יופיעו בהן בעיקר FIN.
  - לעומת זאת, דפדפנים (Chrome, Firefox) נוטים לפתוח ולסגור חיבורים בתדירות גבוהה, מה שמוביל ליותר RST.

- **שימושים מעשיים:**

התפלגות אירועי סיום חיבורים יכולה לשמש כמאפיין חשוב בזיהוי אפליקציות, גם כאשר התעבורה מוצפנת. אפליקציה שמציגה יחס גבוה של RST ככל הנראה אינה שירות סטרימינג, בעוד שאפליקציה שבה כמעט כל הסיומים הם FIN מצביעה על שימוש תקני ומתמשך.

"RTT Distribution"



הגרף שלעיל מציג את **התפלגות זמני הסיבוב (RTT – Round Trip Time)** של חבילות תקשורת שנשלחו וחזרו מהמחשב, פר אפליקציה.

ציר X - מתאר בשניות את זמני ה- RTT.

ציר Y – תדירות המדידה(כמה פעמים נמדד ערך מסוים של RTT).

- **מה הגרף מציג ?**הגרף מציג את ההתפלגות (היסטוגרמה) של זמני ה-RTT לכל אפליקציה. בכל עמודה (Bin) אנו רואים כמה חבילות נמדדו עם RTT באותו טווח זמן ספציפי לדוגמא בטווח 0.05 עד 0.10 יש כמות גדולה של RTT על כמה אפליקציות במקביל.

מהו?

RTT

- (Round Trip Time) RTTהוא פרמטר קריטי במדידת ביצועי רשת. הוא מייצג את הזמן שלוקח לחבילה:

  1. לצאת מהמחשב שלך לשרת המרוחק.
  2. להיענות מהשרת ולחזור אליך.

RTT נמוך מעיד על תקשורת מהירה ויעילה – ולכן הוא חשוב במיוחד עבור שירותים כמו שיחות וידאו או סטרימינג.

- **מה ההבדל בין האפליקציות?** ייתכן שאפליקציות כמו Chrome או Firefox מציגות התפלגות RTT נמוכה יותר לעומת שירותי סטרימינג כמו YouTube או Zoom, שבהם ייתכן שהרשת עוברת מרחקים ארוכים יותר או נתונה לעומסים גבוהים.

- **למה זה קורה?** RTT מושפע ממיקום השרתים, איכות החיבור ומהירות העיבוד. לדוגמה, דפדפנים עשויים להתחבר לשרתים קרובים (או להשתמש בזיכרון מטמון), בעוד ששירותי סטרימינג עשויים לשרת לקוחות ממיקומים שונים.
- **מה אפשר ללמוד מהגרף?**
  - אפשר לראות אילו אפליקציות מתאפיינות ב־ RTT נמוך ואילו ב־ RTT גבוה יותר.
  - התפלגות ה־ RTT יכולה להעיד על ביצועים של הרשת והשרתים שאליהם מתחברים, RTT נמוך כלומר הקשר חזק ומהיר, RTT גבוה מעיד על מיקום רחוק של השרת.
  - סממן עבור דפדפנים (Chrome, Firefox) בשל העבודה שהם מציגים RTT מגוון, עקב גישה למספר גדול של אתרים ומשאבים – חלקם מקומיים, חלקם רחוקים.
  - אפליקציות סטרימינג כמו YouTube ו־Spotify שומרות על RTT נמוך ועקבי – עקב שימוש נרחב ב־CDN אשר מהווה מערכת להפחתה של RTT ושיפור ביצועים במיקוד חיבורים לשרתים קבועים.
  - Zoom מציג פיזור רחב יותר, תוצאה אפשרית של חיבורי זמן-אמת שצריכים להסתגל לרשתות שונות ולמשתמשים מכל העולם.

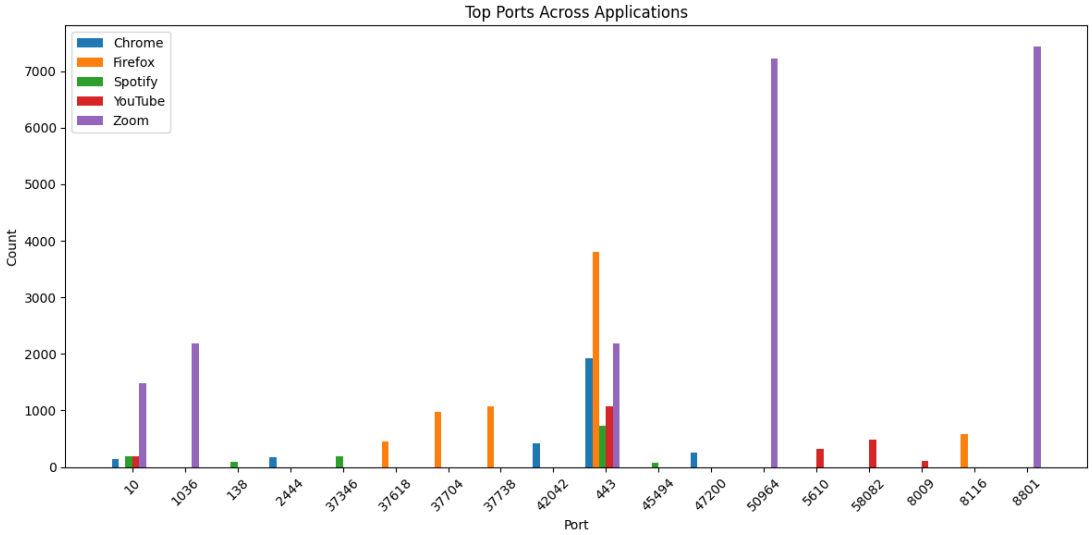
שימושים מעשיים:

- ניתוח RTT מאפשר **סיווג אפליקציות גם כאשר התוכן מוצפן** ,על סמך פרופיל הביצועים.
- ניתן לאתר ביצועי רשת לקויים, נקודות עומס או שרתים מרוחקים לפי התפלגות ה־ RTT.
- בעת אבטחת מידע או ניתוח אנונימיות RTT – ניתן לחשוף **איזו אפליקציה הייתה בשימוש** מבלי לדעת את התוכן המועבר.

אפליקציה	מאפייני RTT בולטים	פרשנות
Chrome	2 שיאים בולטים סביב 0.02 ו־0.08 שניות, עם פיזור רחב כולל זנב עד 0.33	שימוש בשרתים שונים, כנראה תוצאה של טעינת משאבים מאתרים שונים, חלקם קרובים וחלקם רחוקים יותר.
Firefox	פיזור דומה ל Chrome-אך מעט יותר צפוף בטווחים קצרים יותר	מצביע על גישה דומה מבחינת שרתים, אך ייתכן שפעולות הטעינה נעשו על פחות משאבים או אתרים ממוקדים יותר.

Spotify	ריכוז חד סביב 0.03–0.05	מעיד על תקשורת יציבה עם שרתים קבועים וממוקמים קרוב) כמו שרתי סטרימינג של.(Spotify
YouTube	ריכוז גבוה בטווח של 0.01–0.05	תואם לסטרימינג יעיל.
Zoom	מעט פיזור גבוה יותר (0.01 עד 0.2)	שיחות וידאו דורשות חיבור רציף ולכן פיזור ה־ RTT משקף גם עומסי רשת זמניים וגם גישה לשרתים גלובליים במידת הצורך.

"Top Ports Across Applications"



הגרף מציג את **הפורטים הפעילים ביותר** עבור כל אחת מהאפליקציות שנבדקו, Chrome : Firefox, Spotify, YouTube.ו-Zoom-

-ציר הX- מתאר את מספר הפורטים לדוגמא (HTTP Port 80).

-ציר הY- מתאר את מספר החבילות שנשלחו או התקבלו דרך כל פורט.
בציר ה־ Xמופיעים מספרי פורטים (Ports), ובציר ה־ – Yמספר החבילות שנשלחו/התקבלו דרך כל פורט עבור האפליקציה הרלוונטית.

מהו פורט?

Port - הינו מזהה מספרי בתוך פרוטוקול התקשורת TCP או UDP אשר משמש לניתוב חבילות לאפליקציה או שירות מסוים.
דוגמאות לפורטים נפוצים:

- **מה ההבדל בין האפליקציות ?**גרף זה מציג אילו פורטים נמצאים בשימוש עבור כל אפליקציה. למשל, אפליקציות מסוימות עשויות להשתמש בעיקר בפורט 443 (HTTPS)בעוד שאחרות ייתכן וישתמשו גם בפורטים נוספים לצרכים שונים (כמו פורטים שמשויכים ליישומי תקשורת ספציפיים).
- **למה זה קורה ?**ההבדלים נובעים מהפרוטוקולים והשירותים שכל אפליקציה תומכת בהם. לדוגמה, אפליקציות וידאו כמו YouTube או Zoom עשויות לעבור תעבורה מוצפנת דרך פורטים סטנדרטיים (כמו 443) ואילו אפליקציות אחרות) כמו Spotify) עשויות להפעיל חיבורים לשרתים שדורשים פורטים נוספים לתקשורת ייעודית.

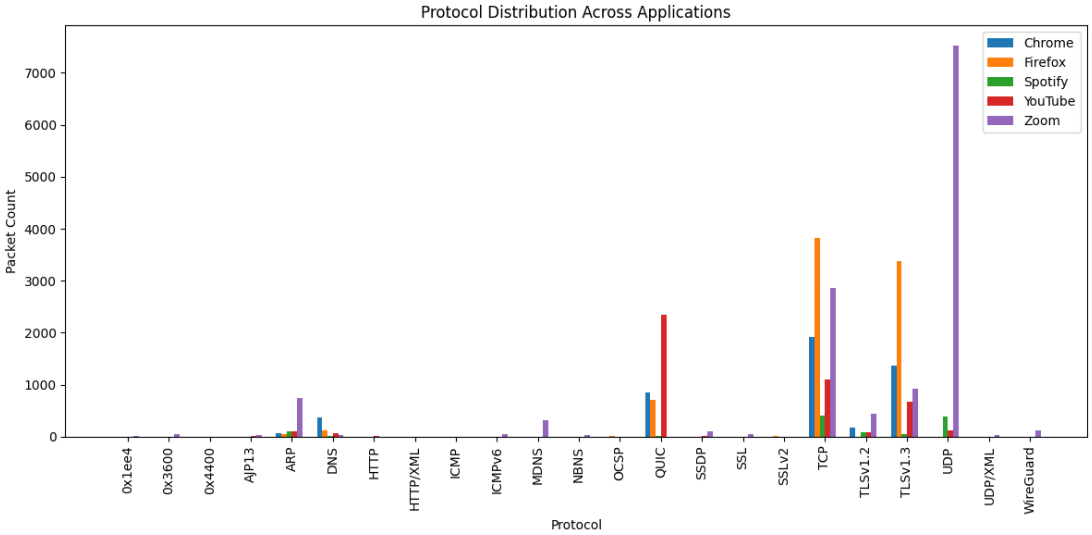
- **מה אפשר ללמוד מהגרף?**
  - הפורטים הפעילים ביותר בכל אפליקציה.
  - הבדלים בין אפליקציות שונות למשל יו-טיוב הינו משתמש בפורט 443, כלומר הוא משתמש בפרוטוקול HTTPS או QUIC בעוד שאפליקציות אחרות עשויות להשתמש בפורטים שונים.
  - ניתן לזהות פורטים חריגים או לא סטנדרטיים שאפליקציות מסוימות פותחות.
  - פורט 443 הוא המשותף לרוב האפליקציות – הוא פורט תקני ל־HTTPS ומספק תעבורה מוצפנת.
  - Zoom הינה אפליקציית יוצאת דופן עקב שימוש מסיבי בפורטים ייעודיים ולא סטנדרטיים, דבר שמקל על זיהוי האפליקציה גם ללא גישה לתוכן.
  - Spotify ו־YouTube מציגות חתימות פורטים מגוונות, כנראה עקב תשתית CDN המערכת אשר עוזרת בהפחתת ה-RTT או פרוטוקולים שונים שנעשה בהם שימוש.

שימושים מעשיים:

- ניתוח פורטים יכול לשמש **לזיהוי סוג אפליקציה גם כאשר הנתונים מוצפנים** ,ע"י זיהוי פורטים ייחודיים או דפוסי שימוש.
- **Zoom למשל קל מאוד לזיהוי** ,בשל השימוש הכבד בפורטים לא רגילים.
- פורטים חריגים עשויים גם להעיד על תקשורת חשודה, ולכן חשוב במערכות ניטור ואבטחה.

אפליקציה	פורטים בולטים	פרשנות
Chrome	443, 42042, 1036	שימוש חזק ב-443 מעיד על גלישה מוצפנת ,(HTTPS) כנראה דרך אתרים שונים.
Firefox	,443טווח 37618–37738	גם כאן ניכר שימוש נרחב ב-443 ,(HTTPS) .
Spotify	מגוון פורטים, כולל 2444, 37704	מעיד על תקשורת עם שרתים ספציפיים.
YouTube	בעיקר 443, עם פעילות ב-5610, 5802	YouTube עושה שימוש ב-HTTPS להעברת וידאו.
Zoom	פורטים בולטים: 8801, 59064, 42042	Zoom מפעילה תעבורה על גבי פורטים ייעודיים, אשר עשויים להשתנות בהתאם לאזור ולסוג השירות.

"Protocol Distribution Across Applications"



גרף זה מציג את פריסת הפרוטוקולים בהם נעשה שימוש בפועל על ידי אפליקציות שונות : Chrome, Firefox, Spotify, YouTube.Zoom-

-בציר הX – שמות הפרוטוקולים.

-בציר הY – מספר החבילות שנשלחו בכל פרוטוקול.

מהו פרוטוקול?

פרוטוקול תקשורת הוא סט חוקים שמגדיר כיצד מועבר מידע בין מחשבים.
ישנם פרוטוקולים שונים למטרות שונות:

- **מה ההבדל בין האפליקציות?** ייתכן שתראו שאפליקציות מסוימות מציגות שימוש גבוה בפרוטוקולים כמו TLS,TCP,UDP – פרוטוקול תעבורה המוודא הצפנה,זיהוי וחיבור אמין בין לקוח לשרת.
- **למה זה קורה ?**כל אפליקציה מתוכננת למטרה שונה – לדוגמה, דפדפנים ויישומי סטרימינג משתמשים בפרוטוקולים שמבטיחים אמינות ואבטחה ,(TCP/TLS) בעוד ששירותי וידאו חיים מעדיפים מהירות וזמני תגובה נמוכים שעשויים לדרוש.
- **מה אפשר ללמוד מהגרף?**
  - אילו פרוטוקולים דומיננטיים בכל אפליקציה למשל, אם אפליקציה עושה שימוש רב בפרוטוקול TLS ככל הנראה האפליקציה מוצפנת.
  - אפליקציות סטרימינג עשויות להשתמש בפרוטוקולים ייעודיים כגון QUIC או בפרוטוקולים מסוימים לשידור מדיה.
  - אפליקציות מסוימות עשויות להעדיף UDP על פני TCP או להפך.

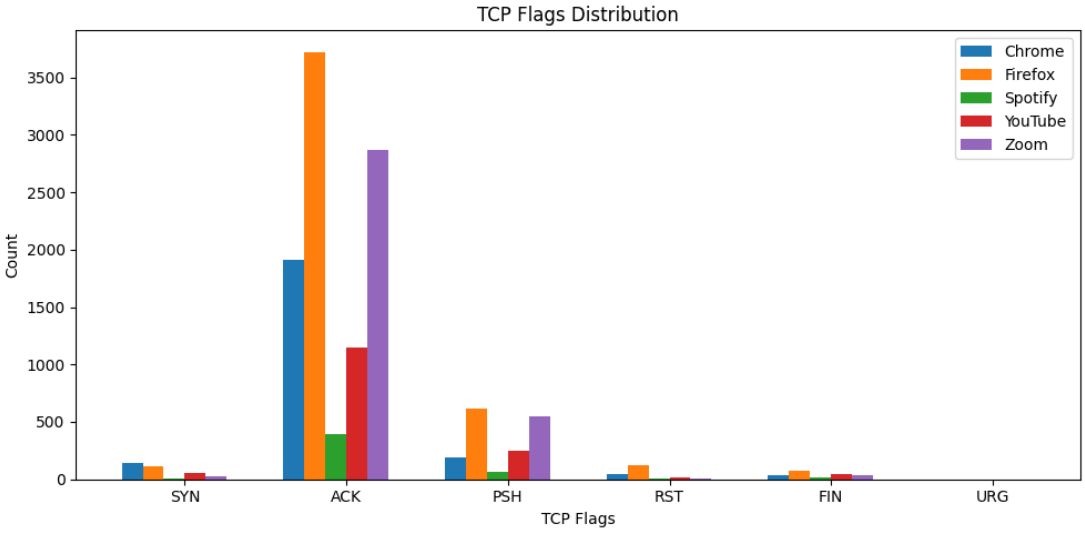
- דפדפנים (Chrome, Firefox) משתמשים בעיקר ב-TLS ו-TCP, אך Chrome בולט בשימוש ב-(QUIC (UDP.
- Zoom מובילה באופן ברור בשימוש ב־UDP, מה שמעיד על תעבורה מותאמת לשיחות וידאו בזמן אמת.
- Spotify ו־YouTube משתמשות בפרוטוקולים מוצפנים כמו TLS ו-QUIC, עם דגש על יציבות והספקה מהירה של תוכן.

שימושים מעשיים:

- ניתן לזהות אפליקציות על פי פרופיל הפרוטוקולים שהן מפעילות, גם כאשר התוכן עצמו מוצפן.
- אפליקציה עם שימוש כבד ב־ UDPכנראה קשורה לשידור חי או קול/וידאו.

אפליקציה	פרוטוקולים עיקריים	פרשנות
Chrome	UDP, TLS1.3, TCP, QUIC	שימוש מאסיבי ב UDP- וב- TLS1.3-מעיד על שימוש בפרוטוקולים מהירים ומוצפנים כמו QUIC, שנפוץ ב-Chrome.
Firefox	TLS1.2, TCP, TLS1.3, UDP	תומך בגרסאות TLS שונות. הדומיננטיות של TCP מצביעה על תקשורת מבוססת חיבור, אך עדיין מופיע.
Spotify	בעיקר TLS1.2 ו TCP, -מעטUDP	סטרימינג אודיו דורש תקשורת יציבה ולכן מתבסס על TCP, עם הצפנה ב TLS.
YouTube	QUIC, TLS1.3, TCP	YouTube משלב בין QUIC שמתבסס על UDP לבין TCP מאובטח, מה שתואם את המעבר ההדרגתי של Google ל-QUIC/HTTP3.
Zoom	שימוש נרחב ב UDP, -TLS, NBNS, ARP	Zoom מתבסס בצורה משמעותית על – UDP בגלל הדרישה לביצועים גבוהים בזמן אמת (וידאו, קול).

"TCP Flags Distribution"



הגרף מציג את **התפלגות דגלי** (TCP Flags) **TCP** שנמצאו בחבילות של אפליקציות שונות Chrome, Firefox, Spotify, YouTube ו-Zoom

ציר X: סוגי הדגלים.

ציר Y: כמות הפעמים שכל דגל הופיע, דהיינו כמה חבילות כללו אותו.

- **מה זה דגלי TCP?** בכל חבילת TCP ישנם ביטים (Flags) המסמנים מצבים שונים, למשל:
  - **SYN** פתיחת חיבור (synchronize).
  - **ACK** אישור קבלה.(acknowledgement).
  - **PSH** דחיפה (push) של נתונים מיידית ליישום.
  - **RST** איפוס (reset) של החיבור.
  - **FIN** סיום (finish) חיבור TCP.
  - **URG** מציין נתונים בעדיפות גבוהה יותר.
- **מה ההבדל בין האפליקציות?** בהסתכלות על הדגלים, ייתכן שאפליקציות עם אינטראקציות מרובות (כמו דפדפנים) יראו יותר חבילות עם דגל SYN (פתיחת חיבור) ו-ACK, לעומת אפליקציות שמבצעות חיבורים ארוכים ויציבים (כמו סטרימינג), שבהן ייתכן שתראו יותר ACK ואולי פחות SYN, FIN או RST.

- **למה זה קורה?** ההבדלים נובעים ממבנה התקשורת: אפליקציות הדורשות חיבורים תכופים וחד-פעמיים (למשל טעינת אתרים) יצרו יותר חיבורים חדשים (SYN), בעוד ששירותי סטרימינג שמתחברים לשרת לאורך זמן יציגו פחות פתיחות וסגירות חיבור.

- **מה אפשר ללמוד מהגרף?**
  - כמה חיבורים נפתחו (SYN) וכמה נסגרו.(FIN/RST)
  - האם יש הרבה RST (איפוס חיבור) שמצביע על ניתוקים פתאומיים.
  - כמה ACK נשלחים ביחס לפרוטוקול האפליקציה) צפוי שיהיו הרבה ACK ברוב האפליקציות.
  - כל האפליקציות מציגות נוכחות גבוהה מאוד של ACK – ביט תקני בתקשורת TCP.
  - Firefox ו-Zoom מציגות גם הרבה RST, מה שמרמז על ניתוקים פתאומיים.
  - Spotify ו-YouTube מתאפיינות בתעבורת ACK ו-PSH עם מעט מאוד פתיחות/סגירות חיבורים, מה שמעיד על שימוש בחיבורים ארוכי טווח.
  - כמעט אין שימוש בדגל URG, כפי שמצופה באפליקציות מודרניות.

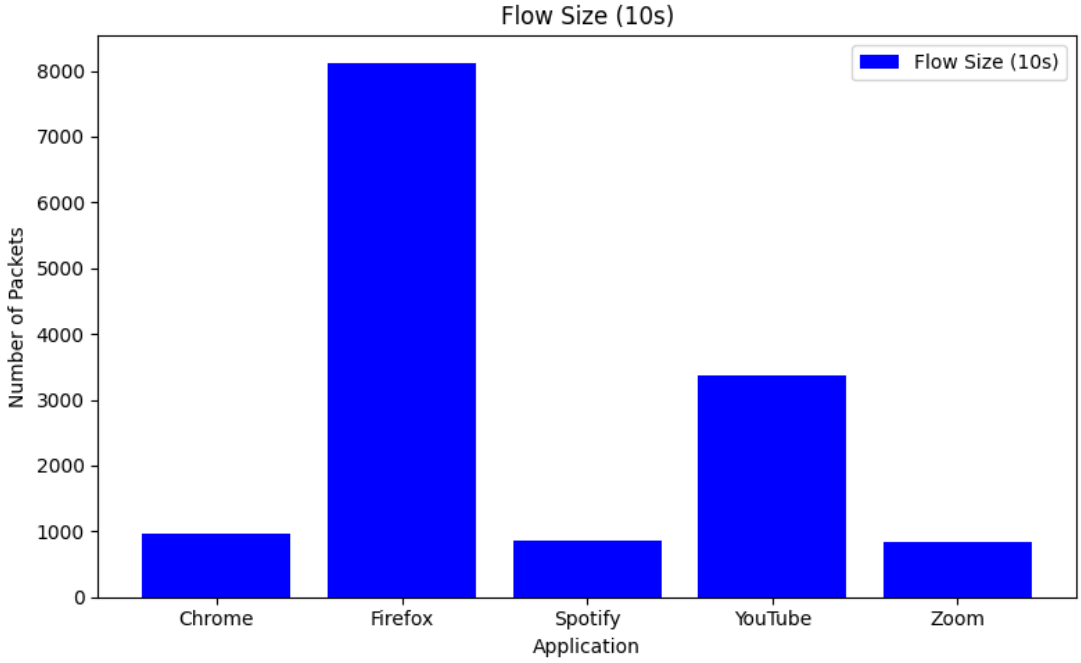
שימושים מעשיים:

- ניתוח דגלי TCP מספק תובנות על **אופי התעבורה** של אפליקציה:
  - האם היא פתחה הרבה חיבורים SYN → ?גבוה
  - האם יש הרבה ניתוקים לא תקניים RST → ?גבוה
  - האם מדובר בחיבור מתמשך ACK → ?ו PSH-שולטות
- ניתן לנצל חתימת הדגלים לזיהוי סוג אפליקציה גם כשהתוכן מוצפן.

אפליקציה	דגלים דומיננטיים	פרשנות
Chrome	מעט ACK, PSH, FIN, RST	כמות גדולה של ACK מצביעה על תעבורה אינטנסיבית PSH .מרמז על דחיפת תוכן מיידית לדפדפן.
Firefox	ACK באופן מובהק , PSHוRST-	מראה דפוס דומה ל- Chrome אך עם יותר RST אשר מעיד על סגירות חיבור פתאומיות.
Spotify	מעט מאוד ACK, PSH, SYN/FIN	מצביע על חיבור יציב ומתמשך – פחות

		פתיחות או סגירות חיבור, ויותר תעבורת ACK רציפה.
YouTube	ACK, PSH, מעט FIN	גם כאן אנו רואים פעילות רבה לאורך חיבור קיים, עם פחות סגירות.
Zoom	ACK , PSH, גם מעט -FINוRST	Zoom מציג יחסית הרבה– ACK עקב צורך בתקשורת דו-כיוונית רציפה. גם RST מופיע, אולי בשל ניתוקים פתאומיים.

"Flow Size(10s)"



גרף זה מציג את **מספר החבילות (packets)** שהועברו בכל אפליקציה במהלך **10השניות הראשונות** של השימוש בה.

ציר X- מופיעות אפליקציות

ציר Y- מספר החבילות שנשלחו או התקבלו.

מה זה Flow Size?

- **Flow Size-** הוא מדד לכמות החבילות שנשלחו בפרק זמן נתון (כאן: 10 שניות).
- הוא מתאר עד כמה האפליקציה "פעילה" או מייצרת תעבורה עם תחילת השימוש.
- שונה מ־ Flow Volume(שמודד את סך הבייטים), כאן מדובר רק במספר החבילות.

  - **מה מציג ?** מספר החבילות (Flow Size) וסך נפח התעבורה (Flow Volume) ב־10 השניות הראשונות של כל אפליקציה.
  - **מה זה אומר ?** מראה עד כמה האפליקציה מייצרת/מקבלת תעבורה באופן מהיר בהתחלת הסשן שלה Flow Size .הוא מספר החבילות Flow Volume , הוא סך הבתים שנשלחו/התקבלו.
  - **מה ההבדל בין האפליקציות?** אפליקציות כמו YouTube או Zoom עשויות להראות ערכים גבוהים יותר של "flow volume" (נפח נתונים) ב-10 השניות הראשונות לעומת דפדפנים, שמבצעים מספר רב של חיבורים קצרים אך עם נפח נתונים נמוך יותר.

- **למה זה קורה?** סטרימינג וידאו דורש העברת נתונים בכמויות גדולות (buffering, שידור וידאו), בעוד שדפדפנים טוענים דפי אינטרנט שיכולים לכלול מספר רב של חיבורים קטנים.

- **מה אפשר ללמוד?**

  - האם האפליקציה מייצרת "פרץ" גדול של תעבורה בהתחלה) כמו סטרימינג וידאו שעושה .
  - אילו אפליקציות עושות הרבה התקשרויות קצרות או ממושכות כבר בשניות הראשונות.
  - Firefox מראה את פרץ התעבורה החזק ביותר ב־10 השניות הראשונות – מצביע על צריכה מרובה של משאבים.
  - YouTube גם כן מראה זרימה גבוהה יחסית – עקב צורך בהעברת תוכן וידאו.
  - Spotify ו-Zoom שומרים על קצב מתון בשלב ההתחלתי.
  - Chrome נמצא באמצע – פחות מ־YouTube/Firefox, יותר מ־Spotify/Zoom

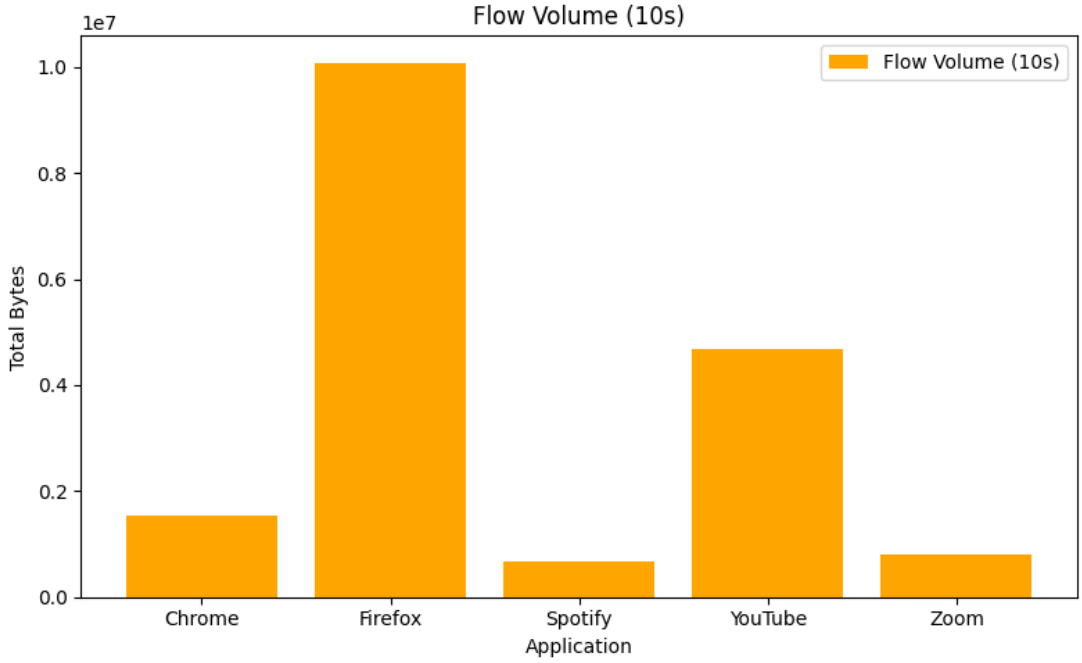
מה אפשר ללמוד מזה?

- אפליקציות שמבצעות "burst" גדול של חבילות בתחילת ההפעלה ניתן לזהות לפי Flow Size גבוה.(YouTube, Firefox)
- Flow Size יכול לשמש לאיתור תבניות שימוש ייחודיות – גם כשאין גישה לתוכן עצמו (אנונימיות/הצפנה).
- אפשר להשתמש במדד זה לבניית **חתימה לזיהוי אפליקציות לפי אופי השימוש הראשוני**.

אפליקציה	מספר חבילות	פרשנות
Firefox	מעל 8000 חבילות	פעילות עצימה מאוד – כנראה טעינה של דף מורכב עם משאבים מרובים. ייתכן שמדובר בגלישה לאתר כבד במיוחד.
YouTube	כ-3400 חבילות	משקף התחלה של סטרימינג וידאו – ייתכן בתחילת הסרטון, מה שמסביר את הגידול המהיר בתעבורה.
Chrome	סביב 950 חבילות	פחות אינטנסיבי מ Firefox- –ייתכן שבוצעה גלישה לדף

		קל יותר, או ש Chrome- אופטימלי יותר מבחינת פתיחת חיבורים.
Spotify	כ-900 חבילות	התנהגות יציבה ולא אגרסיבית – סטרימינג שמע דורש פחות משאבים בהשוואה לווידאו.
Zoom	גם סביב 900 חבילות	בשניות הראשונות Zoom , עדיין מקים את החיבור לפני התחלה ממשית של שיחה/וידאו, ולכן פחות חבילות.

"Flow Volume (10s)"



הגרף מציג את **סך נפח הנתונים (Total Bytes)** שהועברו על ידי כל אפליקציה במהלך **10 השניות הראשונות** להפעלתה.

ציר X- שמות האפליקציות.

ציר Y- סך הבתים שנשלחו או התקבלו

מה זה Flow Volume ?

- **Flow Volume-** מודד את נפח התעבורה במונחים של **סך כל הבתים** –כלומר, כמה מידע ממשי עבר דרך הרשת, בלי קשר למספר החבילות.
- זהו מדד חשוב לזיהוי **אפליקציות עם דרישות רוחב פס גבוהות** כמו וידאו, שיתוף מסך וכו.

  - **מה זה אומר ?**מראה עד כמה האפליקציה מייצרת/מקבלת תעבורה באופן מהיר בהתחלת הסשן שלה Flow Size .הוא מספר החבילות Flow Volume , הוא סך הבייטים שנשלחו/התקבלו.
  - **מה ההבדל בין האפליקציות?** אפליקציות כמו YouTube או Zoom עשויות להראות ערכים גבוהים יותר של "flow volume" (נפח נתונים) ב-10 השניות הראשונות לעומת דפדפנים, שמבצעים מספר רב של חיבורים קצרים אך עם נפח נתונים נמוך יותר.

- **למה זה קורה?** סטרימינג וידאו דורש העברת נתונים בכמויות גדולות (buffering, שידור וידאו), בעוד שדפדפנים טוענים דפי אינטרנט שיכולים לכלול מספר רב של חיבורים קטנים.

- **מה אפשר ללמוד?**

  - האם האפליקציה מייצרת "פרץ" גדול של תעבורה בהתחלה כמו סטרימינג וידאו שעושה.
  - אילו אפליקציות עושות הרבה התקשרויות קצרות או ממושכות כבר בשניות הראשונות.
  - **Firefox** יצר את הנפח הגבוה ביותר – תואם ל־ Flow Size הגבוה מהגרף הקודם.
  - **YouTube** הציג נפח משמעותי – תוצאה של העברת וידאו בתחילת הסרטון.
  - **Spotify ו- Zoom** שמרו על נפח תעבורה נמוך בשניות הראשונות – כל אחת מסיבותיה:
  - Spotify- אודיו קל יותר מבחינת נפח.
  - Zoom- ההתחברות הראשונית כוללת פחות מדיה.
  - Chrome- מציג איזון בין ביצועים לנפח – נמוך מ-YouTube אך גבוה מ-Zoom.

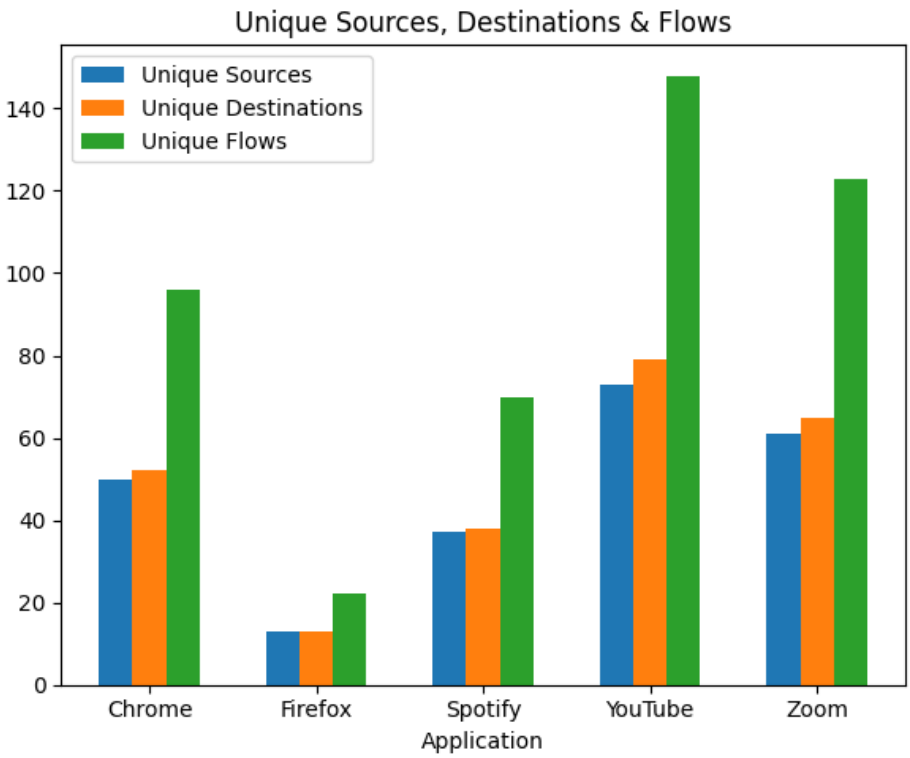
שימושים מעשיים:

- **ניתוח Flow Volume מאפשר לזהות אפליקציות עתירות תעבורה** ,גם אם התוכן מוצפן.
- כשמשווים את Flow Size ל־ Flow Volume ניתן להבין האם האפליקציה שולחת הרבה חבילות קטנות.
- השוואת שני המדדים יחד מספקת חתימה ברורה לזיהוי אפליקציות בצורה עקיפה.

אפליקציה	נפח נתונים כולל	פרשנות
Firefox	מעל 10 מיליון בייט	גבוה ביותר – ייתכן עקב טעינה של אתר כבד מאוד עם הרבה מדיה תומך בממצאי Flow Size.
YouTube	כ־5 מיליון בייט	צפוי – מדובר באפליקציה להעברת וידאו, ולכן עם תחילת סטרימינג נשלחת

		כמות נכבדה של נתונים (buffering).
Chrome	כ־1.5 מיליון בייט	מראה שימוש מתון יחסית – אולי כתוצאה מטעינת אתר קליל יותר או ש Chrome-מצמצם תעבורה בתחילת שימוש.
Zoom	מעט מתחת למיליון בייט	משקף שלב חיבור ראשוני בלבד, לפני התחלת שיחה ממשית או שיתוף מסך.
Spotify	כ־800,000 בייט	נפח נמוך יחסית – צפוי משירות סטרימינג אודיו שבו נדרש פחות מידע בהשוואה לווידאו.

Unique Sources, Destinations & Flows"



גרף זה מציג שלושה מדדים חשובים עבור כל אפליקציה:

- **כתובות מקור ייחודיות – (Unique Sources)** כמה כתובות שונות שלחצו בקשות למכשיר.
- **כתובות יעד ייחודיות – (Unique Destinations)** לכמה כתובות שונות האפליקציה התחברה.
- **זרמים ייחודיים – (Unique Flows)** כמה זרמי תקשורת נפרדים נוצרו (כל זרם כולל כתובת מקור, כתובת יעד ולעיתים גם פורטים).

מהו זרם(Flow)?

- "Flow"הוא **חיבור חד-כיווני או דו-כיווני בין שני קצוות** (source ↔ destination), לרוב מבוסס על חמישייה:
  - IPמקור, פורט מקור
  - IPיעד, פורט יעד
  - פרוטוקול(TCP/UDP)
- ניתוח הזרמים מסייע להבין **אופי וגיוון התעבורה** –האם היא מרוכזת או מבוזרת.

- **מה ההבדל בין האפליקציות?** דפדפנים כמו Chrome או Firefox עשויים להציג מספר גבוה יותר של מקורות ויעדים, כיוון שהם גולשים לאתרים רבים. לעומת זאת, שירותי סטרימינג או שיחות וידאו עשויים להציג זרמים פחות מגוונים (חיבור אחד או מספר מועט של שרתים).
- **למה זה קורה?** דפדפנים מבצעים גישה להרבה אתרים ושירותים בו זמנית, בעוד ששירותי סטרימינג מתמקדים בהעברת תכנים ממקור אחד או מספר מוגבל של מקורות.
- **מה אפשר ללמוד?**
  - כמה מקורות שונים מדברים עם האפליקציה וכמה יעדים שונים היא יוצרת איתם קשר.
  - האם האפליקציה פונה להרבה שרתים (כמו דפדפן שנטען הרבה אתרים), או שמא היא פונה לפחות יעדים (כמו שירות ספציפי אחד)
  - YouTube ו-Zoom מובילות בכמות זרמים ייחודיים – עקב תעבורה מרובת ערוצים.
  - Chrome מפעילה זרמים רבים יחסית, אך פחות מ־YouTube – אולי עקב טעינה מהירה של אתרים מרובי משאבים.
  - Spotify מציגה איזון: יותר מ-Firefox, פחות מ-YouTube.
  - Firefox בולט בפשטות היחסית – אולי נפתח בו עמוד בודד.

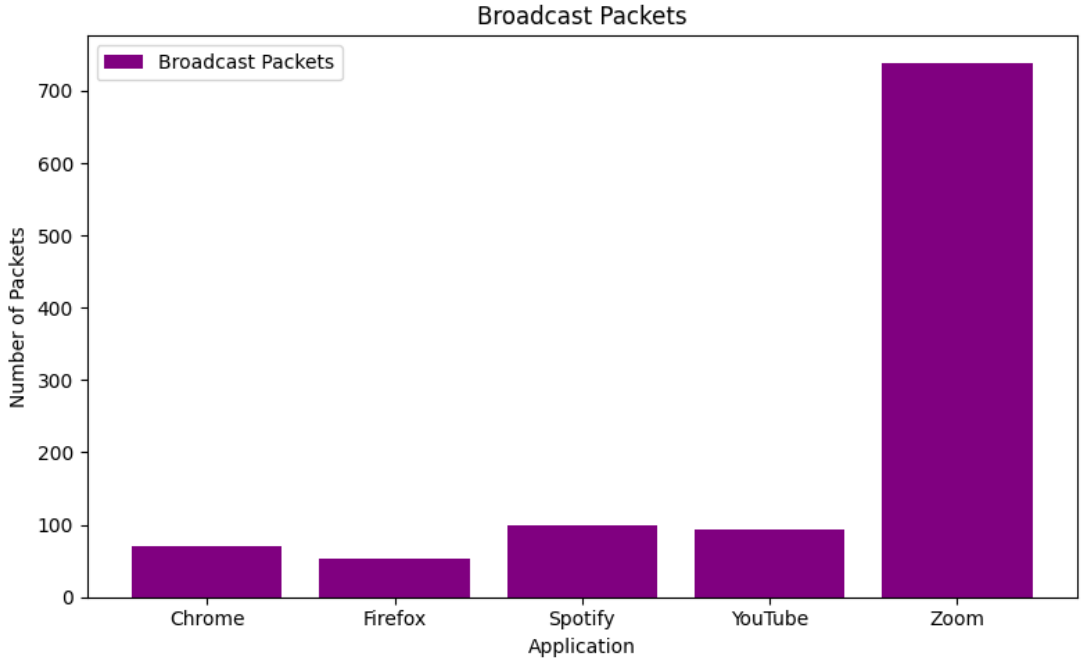
שימושים מעשיים:

- מדדים אלו מסייעים **לזהות את סוג האפליקציה לפי פרופיל הגישה שלה לשרתים**:
  - אפליקציות עם זרמים מגוונים פונות להרבה שרתים – תואם לדפדפנים.
  - אפליקציות עם מספר נמוך של יעדים אך תעבורה מאסיבית – תואם לשירותי סטרימינג או וידאו.
- מועיל לניתוח **אנונימיות או התנהגות חשודה** –לדוגמה, חיבור פתאומי לעשרות כתובות שונות.

אפליקציה	ניתוח ייחודיות כתובות וזרמים	פרשנות
Chrome	מקורות ויעדים בקירוב: 50 זרמים בקירוב: 95	מצביע על פעילות דפדפנית טיפוסית – גישה למספר אתרים/שירותים בו זמנית
Firefox	נתונים נמוכים יחסית	ייתכן שהאינטראקציה הייתה קצרה או ממוקדת באתר בודד.
Spotify	כ-37 מקורות/יעדים, כ-70 זרמים	מגוון חיבורים יחסית רחב – ייתכן שימוש בכמה שרתים

		אשר מבצעים ניתוחי ניתוב מוזיקה שונים.
YouTube	72 מקורות, 79 יעדים, 150 זרמים בקירוב	תעבורה עשירה מאוד – כנראה כוללת וידאו, המלצות, תגובות, פרסומות. זרמים רבים תואמים גישה רחבה לשרתים שונים.
Zoom	60 מקורות, 65 יעדים, 125 זרמים בקירוב	נפח גבוה של זרמים – מעיד על פעילות תקשורת מתמשכת ובמקביל.

"Broadcast Packets"



גרף זה מציג את **מספר חבילות השידור (Broadcast)** שנשלחו או התקבלו במהלך פעילות של כל אחת מהאפליקציות Chrome, Firefox, Spotify, YouTube ו-Zoom :

ציר X – מציג את שם האפליקציה.

ציר Y – מספר החבילות שנשלחו מקומית.

מה זה Broadcast ?

- **Broadcast -** הינה טכניקת שידור בה החבילה נשלחת **לכל המחשבים ברשת המקומית**.
- השימוש ב Broadcast- נפוץ בפרוטוקולים כמו – ARP, NBNS, SSDP ובעיקר כאשר המכשיר צריך לגלות התקנים אחרים ברשת.

  - **מה מציג ?**מספר החבילות שנשלחו בכתובות Multicast ו־ Broadcastעבור כל אפליקציה.
  - **מה זה Multicast ו־ Broadcast** :

Broadcast:

1.) שולח חבילות לכל המשתמשים ברשת המקומית.
2.) משתמש בכתובת MAC מיוחדת.

Multicast :

1.) שולח חבילות למשתמשים ספציפיים.

2.) משתמש בכתובות MAC Multicasting .

- **מה ההבדל בין האפליקציות?** יתכן שאפליקציות שמבצעות שידורים קבוצתיים או הודעות רשת (למשל, אפליקציות וידאו או שירותי עדכונים) יציגו ערכים גבוהים יותר של חבילות Broadcast או Multicast לעומת אפליקציות אינדיבידואליות כמו דפדפנים.
- **למה זה קורה?** שימוש ב-Multicast/Broadcast נדרש כאשר המידע צריך להיות מופץ למספר נמענים בו-זמנית – דבר שכיח בשידורי וידאו או עדכוני רשת בזמן אמת.
- **מה אפשר ללמוד?**

  - עד כמה האפליקציה משתמשת בהודעות לכלל הרשת (Broadcast) או לקבוצה מסוימת.(Multicast)
  - Zoom בולט באופן קיצוני בשימוש ב-Broadcast – מובהק עבור אפליקציות בזמן אמת הדורשות גילוי וניהול תקשורת מתקדמת.
  - דפדפנים מבצעים Broadcast רק בעת צורך.
  - אפליקציות סטרימינג כמו YouTube ו־Spotify משתמשות ב-Broadcast רק בשלבים ראשוניים, ולא לאורך כל הסשן.

אפליקציות/VoIP וידאו/שידור יכולות להשתמש ב־ Multicast להעברת תעבורה למספר נמענים בו־זמנית.(VoIP - הינו שיטה להעברת שיחות טלפון באמצעות כתובות IP ושימוש בפרוטוקולי תקשורת במקום קווי טלפון רגילים).

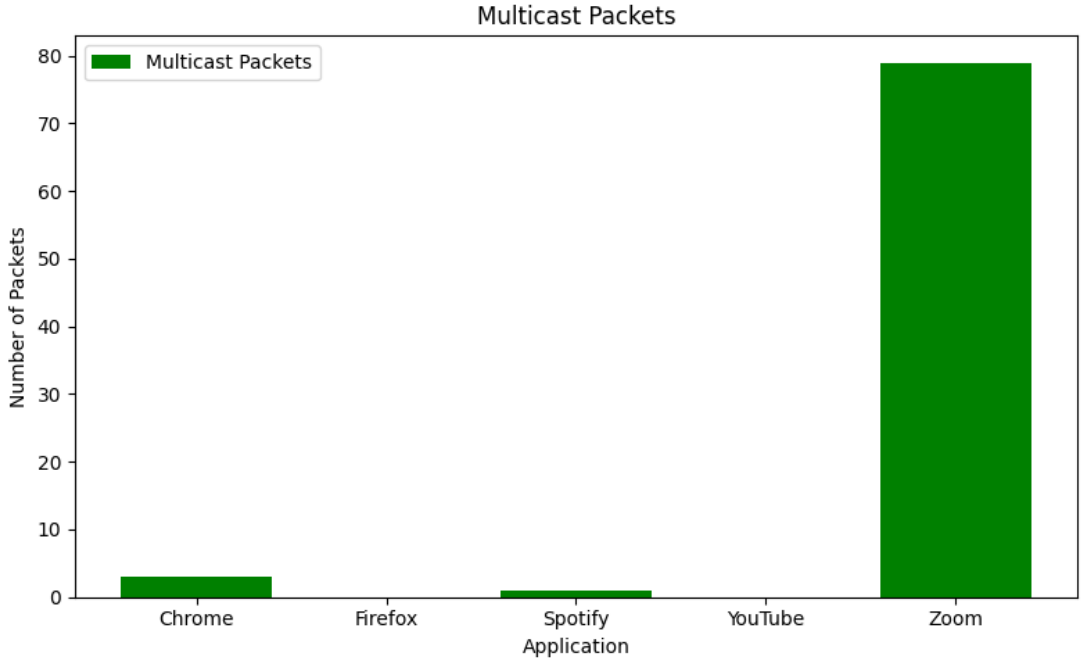
שימושים מעשיים:

- מספר גבוה של Broadcast יכול להוות **חתימה לזיהוי אפליקציה** ,גם כאשר התוכן עצמו מוצפן.
- ניתן לאתר **אפליקציות תקשורת חיה**.
- במערכות ניתוח תעבורה – כמות חריגה של Broadcast עשויה להעיד על אפליקציות כבדות, פעילויות לא רגילות או בעיות קונפיגורציה.

אפליקציה	מספר חבילות Broadcast	פרשנות
Zoom	גבוה מאוד (מעל 700)	מבצע פעולות חשיפה של משתמשים נוספים עבור מכשירים שנמצאים בקרבת מקום.
Spotify / YouTube	סביב 90–100 חבילות	אפליקציות אלה ככל הנראה שולחות מספר הודעות גילוי בתחילת סטרימינג או חיבור לשרתים – אך לא תלויות ב-Broadcast ברמה גבוהה.

Chrome / Firefox	סביב 50–70 חבילות	שימוש מתון – בדפדפנים מדובר לרוב בשימוש של פרוטוקול TCP וחיבור לשרת באמצעות בקשת DNS

"Multicast Packets"



הגרף מציג את **מספר חבילות ה Multicast-**שזוהו עבור כל אחת מהאפליקציות ,Chrome : Firefox, Spotify, YouTube ו – Zoom:

ציר X – מופיעות האפליקציות

ציר Y – מספר החבילות multicast שהועברו..

מהו?Multicast

- **Multicast-** היא טכניקת שידור בה נשלחת חבילה לקבוצת מחשבים מוגדרת מראש ולא לכולם כמו ב־ Broadcast
- שימושי במיוחד לשידורים קבוצתיים, כמו שיחות ועידה, סטרימינג חי, וכו.

- **מה ההבדל בין האפליקציות?** יתכן שאפליקציות שמבצעות שידורים קבוצתיים או הודעות רשת (למשל, אפליקציות וידאו או שירותי עדכונים) יציגו ערכים גבוהים יותר של חבילות Broadcast או Multicast לעומת אפליקציות אינדיבידואליות כמו דפדפנים.
- **למה זה קורה?** שימוש ב-Multicast/Broadcast נדרש כאשר המידע צריך להיות מופץ למספר נמענים בו-זמנית – דבר שכיח בשידורי וידאו או עדכוני רשת בזמן אמת.
- **מה אפשר ללמוד?**

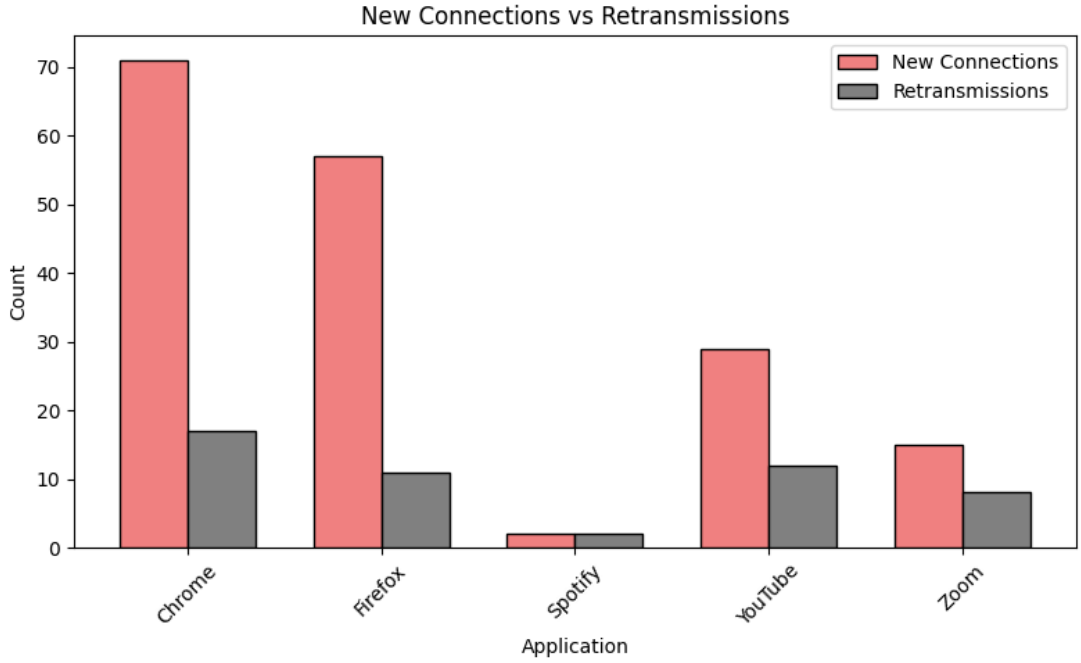
- עד כמה האפליקציה משתמשת בהודעות לכלל הרשת (Broadcast) או לקבוצה מסוימת.(Multicast)
- Zoom היא היחידה שמשתמשת בצורה ברורה ופעילה ב-Multicast, מה שמעיד על התנהלות תקשורת מרובת משתתפים בזמן אמת.
- דפדפנים ואפליקציות סטרימינג (כמו Chrome, Firefox, YouTube) אינם דורשים Multicast – התעבורה אצלם מבוססת על קשרי לקוח-שרת.

שימושים מעשיים:

- **Multicast הוא אינדיקטור חזק לאפליקציות זמן אמת** ,ולכן אם מזוהה שימוש רב בו – ניתן להסיק שמדובר בשירות שיחות או שיתוף.
- ניתן לשלב את המידע עם Broadcast לזיהוי מדויק של אפליקציות כמו Zoom.
- גם כאשר כל התוכן מוצפן – דפוסי Multicast יכולים להסגיר את סוג השירות בשימוש

אפליקציה	שימוש ב-Multicast	פרשנות
Zoom	כ-80 חבילות	שימוש משמעותי Zoom – ככל הנראה משתמש בפרוטוקולים כדי לבצע **גילוי שירותים** ברשת המקומית או ניהול שיחות ועידה עם משתתפים מרובים. מעיד על אופי תקשורת חי ויעיל.
Chrome	כ-3 חבילות	שימוש בסיס ב- DNS
Spotify	1–2חבילות	ייתכן שימוש רגעי בפרוטוקול גילוי כלשהו בעת התחברות.

"New Connections vs Retransmissions"



גרף זה מציג את מספר **החיבורים החדשים (New Connections)** ואת כמות **השידורים החוזרים (Retransmissions)** שבוצעו על ידי כל אפליקציה במהלך פעילותה.

ציר ה X – כולל את כל האפליקציות.

ציר ה Y – מספר החבילות הרלוונטיות לכל קטגוריה.

ציר ה־ Xכולל את האפליקציות, וציר ה־ Yאת מספר החבילות הרלוונטיות לכל קטגוריה.

מה המשמעות של כל מדד?

- **New Connections -** מייצג את מספר הפעמים שהאפליקציה פתחה חיבור TCP חדש.
- **Retransmissions**: **-** מייצג מקרים בהם חבילות נשלחו מחדש בגלל אובדן, שגיאה או חוסר באישור.(ACK)
- **מה ההבדל בין האפליקציות?** אפליקציות כמו דפדפנים עשויות להראות מספר חיבורים חדשים גבוה (עקב טעינת דפים מרובים) עם מספר נמוך יחסית של Retransmissions, מה שמעיד על רשת יציבה. לעומת זאת, אפליקציות כמו Zoom או YouTube עשויות להציג גם מספר גבוה של חיבורים חדשים וגם Retransmissions, מה שעשוי להעיד על עומס ברשת או חיבורים עם מרחק גדול מהשרת.
- **למה זה קורה?** מספר גבוה של Retransmissions יכול להצביע על בעיות בתעבורה – כגון איבוד חבילות, עיכובים או עומסים ברשת, בעיקר כאשר מדובר בשירותי סטרימינג בו זמנית עם חיבורים חדשים מרובים.

- **מה אפשר ללמוד מהגרף?**
  - כמה חיבורים נוצרים בכל אפליקציה. אם הערך גבוה, האפליקציה מבצעת הרבה התקשרויות.
  - כמות Retransmissions גבוהה יכולה להעיד על בעיות איכות רשת או עומס שגורם לאיבוד חבילות.
  - דפדפנים (Chrome ו-Firefox) מציגים מספר חיבורים גבוה אך מספר נמוך יחסית של Retransmissions – סימן לתקשורת יעילה.
  - Zoom ו-YouTube מציגות יחס גבוה יותר של שידורים חוזרים, המעיד על תעבורה תובענית או תנאים פחות יציבים ברשת.
  - Spotify מראה פעילות רשת מינימלית – טיפוסי לאפליקציה שמתחברת פעם אחת ומזרימה לאורך זמן.

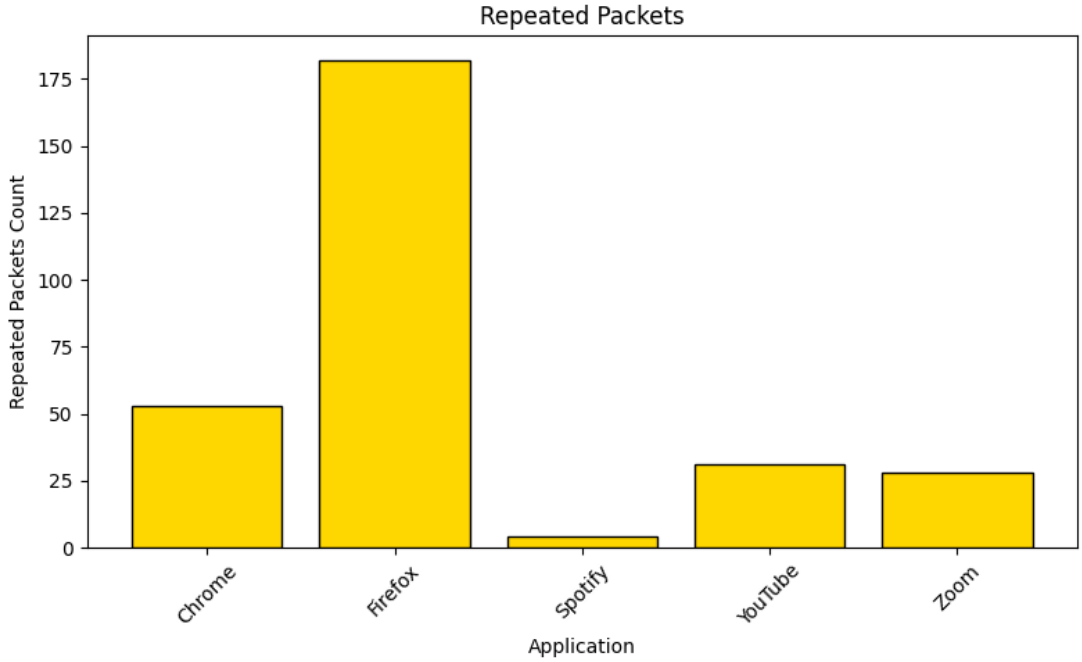
שימושים מעשיים:

- מספר גבוה של **New Connections** יכול להעיד על אפליקציה שנטענת דינמית (כמו דפדפן).
- אפשר להצליב את שני המדדים לצורך **סיווג אפליקציות** גם כאשר כל התוכן מוצפן.

אפליקציה	חיבורים חדשים	שידורים חוזרים	פרשנות
Chrome	כ־71	כ־17	דפדפן מאוד פעיל – יוצר הרבה חיבורים חדשים. מספר השידורים החוזרים לא גבוה מדי כלומר הרשת יציבה.
Firefox	כ־57	כ־11	גם Firefox יוצר חיבורים רבים, פחות מ ,Chrome- עם מעט שידורים חוזרים.
Spotify	כ־2	כ־2	שימוש מועט ביותר – סטרימינג אודיו יציב, חיבור ארוך-טווח, מעט שידורים חוזרים כלומר הרשת תקינה.
YouTube	כ־29	כ־12	פתיחה ממוצעת של חיבורים, אך יחסית הרבה Retransmissions

			–כנראה בגלל נפח נתונים גדול ודרישות סטרימינג שגורמות לעומסים או לאיבודים.
Zoom	כ־15	כ־8	אפליקציית שיחות וידאו דורשת חיבור רציף אך רגישה מאוד לאיכות הרשת ולכן גם במספר חיבורים קטן יחסית, מופיעים שידורים חוזרים.

"Repeated Packets"



הגרף מציג את מספר **החבילות החוזרות (Repeated Packets)** שזוהו במהלך השימוש בכל אחת מהאפליקציות Chrome, Firefox, Spotify, YouTube ו-Zoom.

ציר X – שמות האפליקציות.

ציר Y – מספר החבילות החוזרות כתוצאה מ Retransmissions או Duplicate ACKs

מהן חבילות חוזרות?

- **Retransmission-**: חבילה שנשלחת מחדש כי לא התקבל עליה אישור בזמן (Timeout).
- **Duplicate ACK-**: מתקבל מהצד המקבל כאשר חבילה נעלמה והוא מזהה חוסר ברצף – רמז לבעיה בקו התקשורת.

- **למה זה קורה ?** כאשר הרשת מאבדת חבילות, השולח שולח אותן מחדש כאשר המקבל מזהה חבילה חסרה, הוא עשוי לשלוח Duplicate ACK שמתריע על כך.
- **מה ההבדל בין האפליקציות?** ייתכן שאפליקציות מסוימות, במיוחד אלו שפועלות בסביבות רשת עם איכות פחותה (כמו שיחות וידאו בזמן עומס) יראו ערכים גבוהים יותר של חבילות חוזרות (Retransmissions או Duplicate ACKs) לעומת אפליקציות אחרות שמתחברות לרשת יציבה יותר.
- **למה זה קורה?** ערכים גבוהים של חבילות חוזרות מצביעים על בעיות בתקשורת – כמו איבוד נתונים, שיבושים או עיכובים, מה שעלול לקרות כאשר הרשת עמוסה או כאשר יש איכות חיבור ירודה.

- **מה אפשר ללמוד מהגרף?**
  - כמה בעיות או אובדן חבילות התרחשו בתקשורת של כל אפליקציה.
  - אפליקציה עם ערך גבוה עשויה להיות רגישה יותר לאיבוד חבילות או דורשת מהירות תגובה גבוהה (כמו וידאו חי).
  - Firefox מובילה בבירור עם מספר גבוה מאוד של חבילות חוזרות – מה שמעיד על שיעור גבוה של בעיות רשת/שגיאות.
  - Chrome ו־YouTube מציגות כמות בינונית של שידורים חוזרים, תואם לתנועה הדחופה שלהן.
  - Zoom ו־Spotify שומרות על יציבות מרשימה יחסית – מה שמעיד על מימושים חסכוניים/עמידים לאיבוד חבילות.

שימושים מעשיים:

- מספר חבילות חוזרות יכול לשמש **אינדיקטור לאיכות רשת בזמן אמת**.
- אפליקציות שדורשות אמינות או תקשורת רציפה כמו Zoom/Spotify **המתוכננות לעבודה בתנאים בעייתיים** –ולכן מציגות פחות חזרות.
- **ניתוח חוזר של חבילות** מהווה כלי לזיהוי עומסים, תקלות או התנהגות ייחודית של אפליקציה, גם כשהתוכן מוצפן.

אפליקציה	מספר חבילות חוזרות	פרשנות
Firefox	כ־180 חבילות חוזרות	ערך חריג בגובהו – ייתכן שנגרם מבעיות בתקשורת או עיכובים בתעבורת הרקע. תומך בממצאים קודמים של Retransmissions רבים.
Chrome	כ־50	גם כאן מופיעה כמות לא מבוטלת – תוצאה אפשרית של גלישה לאתרים מרובי משאבים/פרסומות או רשת פחות יציבה.
YouTube	כ־30	תואם את פרופיל הסטרימינג – בו לעיתים נשלחות מחדש חבילות קריטיות לתוכן.
Zoom	כ־27	נתון מפתיע – יחסית נמוך למרות שמדובר בשיחות בזמן אמת. ייתכן ש Zoom- משתמש במנגנונים חכמים לתיקון שגיאות.

סיכום כללי

באמצעות ניתוח הגרפים השונים, ניתן לגבש תמונה רחבה ומעמיקה של **אופן הפעולה של אפליקציות שונות ברשת** ,גם כאשר התוכן עצמו מוצפן. המדדים שנבדקו – כגון מספר חיבורים, כמות תעבורה, שימוש בפרוטוקולים, פורטים, זרמים, חבילות חוזרות RTT ,ועוד מספקים יחד **דפוס התנהגות** ייחודי לכל אפליקציה.

עיקרי הממצאים:

- **דפדפנים (Chrome, Firefox)** נוטים לטעון דפים רבים ומרובי משאבים:
  - יוצרים מספר רב של **חיבורים קצרים**.
  - משתמשים בעיקר ב־**TCP/HTTPS**.
  - תעבורת Broadcast ו Retransmissions-בינונית יחסית.
  - Chromeמשתמש גם בפרוטוקול QUIC המבוסס על UDP.
- **שירותי סטרימינג/וידאו:(YouTube, Zoom)**
  - יוצרים **תעבורה בהיקף גבוה** כבר בשניות הראשונות.(buffering)
  - מציגים זרמים רבים, תעבורה עקבית, ולעיתים גם **Retransmissions** או **חבילות חוזרות** ,במיוחד בזמני עומס.
  - **Zoom** עושה שימוש נרחב ב־ Multicast או Broadcast תואם לתקשורת קבוצתית בזמן אמת.
- **Spotify**:
  - מפעילה **חיבור יציב ומתמשך** להעברת אודיו.
  - מאופיינת בכמות תעבורה נמוכה יחסית, מעט מאוד חבילות חוזרות, ושימוש ממוקד בפרוטוקולים מוצפנים.
  - כמעט ואינה סובלת משידורים חוזרים או בעיות רשת.

מסקנות כלליות:

- ניתן להבדיל בין אפליקציות **לא לפי תוכן התקשורת** ,אלא לפי **אופי השימוש ברשת** :זמני תגובה, פורטים, פרוטוקולים, דגלי TCP, ועוד.

- הנתונים מעידים על **התאמה בין המבנה של האפליקציה לצרכים שלה** – אפליקציות זמן-אמת מתמקדות ביציבות ומהירות, בעוד שדפדפנים גמישים ודינמיים.
- השילוב של מספר גרפים יחד מאפשר **סיווג מדויק של אפליקציות** ,גם כאשר לא ניתן לפענח את המידע המועבר.