פרויקט גמר 2025
רשתות תקשורת

מגישים:

רונן צ'רשניה

רז כהן

שיר ביסמוט

קלרה פרנקו

תוכן עניינים

חלק 1

שאלות

1. A user reports that their file transfer is slow, and you need to analyze the transport layer to identify the potential reasons. What factors could contribute to the slow transfer, and how would you troubleshoot it?
2. Analyze the effects of TCP's flow control mechanism on data transmission. How would it impact performance when the sender has significantly higher processing power than the receiver?
3. Analyze the role of routing in a network where multiple paths exist between the source and destination. How does the path choice affect network performance, and what factors should be considered in routing decisions?
4. How does MPTCP improve network performance?
5. You are monitoring network traffic and notice high packet loss between two routers. Analyze the potential causes for packet loss at the Network and Transport Layers and recommend steps to resolve the issue.

1) אילו גורמים יכולים לתרום להעברת איטית?

- **בקרת עומס ב TCP (Congestion Control):**
TCP בנוי כך שברגע שהוא מזהה אובדן חבילות, הוא מניח שיש עומס ברשת ומאט את קצב השידור. גם אם אין עומס אמיתי, האותות שמתקבלים גורמים לו לפעול בזהירות.

- **חלון הזזה קטן (Sliding Window):**
אם הצד המקבל מאותת שהוא לא יכול לקלוט הרבה נתונים, השולח מאט בהתאם. זה יכול לקרות אם הצד השני עמוס, איטי, או מגביל את גודל הזיכרון לשידור.

- **אובדן חבילות / שידורים חוזרים:**
כל מקרה שבו חבילה "נעלמת" בדרך מאלץ את TCP לשדר אותה מחדש, ומוריד את קצב ההעברה הכולל.

- **זמן השהייה גבוה:**
כשזמן ה- RTT גבוה, TCP שולח חבילות ואז מחכה יותר זמן ל ACK מה שמקטין את קצב ההעברה בפועל.

1) ואיך נפתור אותם?

- **ביצוע ניתוח תעבורה:**
נאסוף את התעבורה גם בצד הלקוח וגם בצד השרת. ונחפש דפוסים של אובדן חבילות, ערכי חלון TCP קטנים, או נפילות מהירות משמעותיות בזרימה.

- **בדיקת RTT וצמיחת חלון העומס (Congestion Window):**
נוכל להשתמש בגרפים של Wireshark כדי לוודא אם החלון גדל באיטיות מוגזמת או מצטמצם לעיתים קרובות בעקבות אובדן מנות.

2) כיצד הדבר ישפיע על הביצועים כאשר לשולח יש עוצמת עיבוד גבוהה משמעותית מזו של המקבל?

- בקרת זרימה ב-TCP נועדה ליצור איזון בין השולח למקבל. הרעיון המרכזי הוא למנוע מצב שבו השולח מעביר נתונים במהירות שהמקבל לא מסוגל לעמוד בה. כאשר יש פערי עיבוד בין השולח (מהיר) למקבל (איטי), הפרוטוקול מגביל את השידור דרך ערך שנקרא Receive Window.

- אם המקבל לא מצליח לעבד את הנתונים בזמן, הוא מאותת לשולח להאט, מה שגורם לכך שהשולח ממתין או משדר בקצב נמוך יותר, למרות שיש לו את היכולת והמשאבים לשדר מהר יותר.

- במקרים קיצוניים, המקבל מודיע על חלון בגודל אפס, והשולח מפסיק את השידור לחלוטין עד לעדכון חדש. התוצאה היא שהביצועים בפועל של הרשת מושפעים מהצד האיטי ולא מהצד החזק יותר.

3) כיצד משפיע בחירת המסלול על ביצועי הרשת?

- **השהייה:**
  נתיבים מסוימים עשויים להיות פיזית ארוכים יותר או עמוסים יותר, מה שגורם לעיכוב גדול יותר.
  בנתיב קצר או פחות עמוס יכולה לשפר את המהירות.

- **רוחב פס (Bandwidth):**
  נתיב עם קיבולת גבוהה מסוגל לשנע יותר נתונים בו זמנית.
  אם נבחר נתיב עם קיבולת נמוכה, עלול להיווצר צוואר בקבוק.

- **אובדן מנות (Packet Loss):**
  נתיבים לא יציבים או כאלו שיש עליהם עומס יתר עשויים לאבד מנות.
  נתיב איכותי יצמצם את הסיכוי לאובדן מידע.

3) ואילו גורמים יש לקחת בחשבון בהחלטות ניתוב?

- **מספר קפיצות (Hop Count)** - כמה נתבים יש בין המקור ליעד.
- **משקל / עלות של קישור (Link Cost)** - ערך שניתן להגדיר ידנית כדי להעדיף קישור מסוים.
- **רוחב פס (Bandwidth)** - לרוב מעדיפים קישורים רחבים ומהירים יותר.
- **השהייה (Delay)** - הזמן שלוקח לנתונים לעבור את הנתיב.
- **אמינות (Reliability)** - האם היו תקלות, שגיאות או אובדן נתונים בעבר.
- **עומס / צפיפות (Congestion)** - יש פרוטוקולים דינמיים שמתחשבים בעומס בזמן אמת.

לסיכום, ניתוב חכם בוחר את הנתיב הטוב ביותר מבין האפשרויות, לפי מדדים שונים שמשפיעים על ביצועים, אמינות וחוויית המשתמש. ברשתות עם ריבוי נתיבים, ניתוב איכותי מבטיח שהמידע יזרום בצורה יעילה, מהירה, ומתאימה לתנאים הקיימים.

4) כיצד פרוטוקול MPTCP משפר את ביצועי הרשת?

MPTCP (Multipath TCP) הוא הרחבה של פרוטוקול ה-TCP, שמאפשר שימוש במספר נתיבים במקביל להעברת תעבורה אחת אחידה. במקום להעביר את כל המידע דרך ערוץ בודד כמו ב-TCP רגיל MPTCP מחלק את הנתונים לכמה תתי-חיבורים העובדים יחד.

שיפור ביצועים ורוחב פס:

- ניתן להשתמש בכמה ממשקי רשת בו זמנית.
- זה מאפשר ניצול טוב יותר של משאבים והגעה למהירויות גבוהות יותר.

עמידות לתקלות:

- אם אחד הנתיבים כושל או נחלש, החיבורים האחרים ממשיכים להעביר מידע.
- המשתמש כמעט ולא ירגיש בתקלה – החיבור נשאר יציב.

איזון עומסים חכם:

- MPTCP יודע לפצל את העומס בין הנתיבים לפי העומס והביצועים של כל אחד מהם בזמן אמת.

5) הסיבות האפשריות לאובדן חבילות בשכבת הרשת

עומס (Congestion) והצפת זיכרון (Buffer Overflow)

בעיה: כאשר כמות התעבורה עולה על יכולת הטיפול של הנתב או על גודל זיכרון שלו, חבילות מושלכות ברגע שהבאפר מתמלא באופן יזום כדי להישאר זמינים.

פתרון: נגדיר מדיניות ניהול עומסים, נשקל להגדיל את רוחב הפס של הקישור או לפצל את העומס על מספר קישורים.

לולאות ניתוב (Routing Loops)

בעיה: טבלאות ניתוב שגויות או הגדרות לא נכונות עלולות ליצור לולאות או מסלולים ארוכים ומסורבלים. במצבים אלו, חבילות עלולות להיזרק עקב פקיעת ערך ה- TTL או לחזור שוב ושוב דרך אותם נתיבים עמוסים.

פתרון: נוודא שהפרוטוקולים מוגדרים נכון. נשתמש ב-Traceroute לזיהוי מסלולים חריגים או לולאות. נבדוק אם יש חפיפות ברשתות (Subnets) או ערכים חסרים בטבלאות הניתוב.

קישורים איטיים או לקויים (Link Congestion / Errors) :
בעיה: חיבורי רשת עמוסים או באיכות ירודה גורמים לירידת חבילות, במיוחד בקישורים פיזיים חלשים (כבלים, סיבים).

פתרון: נבדוק איכות כבלים, נתאים מהירות, נשקול הרחבת רוחב הפס או פיצול עומסים.

5) הסיבות האפשריות לאובדן חבילות בשכבת הרשת
(המשך)

בקרת עומס ב-TCP והעברות חוזרות (Retransmissions)

בעיה: איבודי חבילות גורמים ל-TCP לפרש עומס, להאט שליחה ולייצר עוד העברות חוזרות.
פתרון: ננתח תעבורה (למשל ב-Wireshark) נוודא שהאלגוריתם מתאים לרשת ונבדוק יציבות ללא הפסדים.

אובדן מנות ב-UDP
בעיה: ל-UDP אין מנגנון שליחה חוזרת. איבודי חבילות משמעם איבוד נתונים ישיר ביישום.
פתרון: נטמיע תיקון שגיאות או Retransmission ברמת היישום, נשקול לעבור לפרוטוקול אמין או להוריד קצב.

חוסר איזון במנגנוני Flow Control
בעיה: פער בין קצב השליחה ליכולת הקבלה יוצר הצפת באפרים ואיבוד חבילות.
פתרון: נוודא שהחלון ב-TCP מספיק גדול, נכוונן הגדרות תעבורה ונוודא ניהול זרימה תקין בפרוטוקול.

5) המלצות לפתרון הבעיה

שלב 1: בדיקות בסיס

- להריץ ping ו-traceroute כדי לראות איפה לאורך המסלול מתחילה הבעיה.
- לבדוק אם יש קפיצות זמן פתאומיות או נקודה קבועה שבה מתחילים האובדנים.

שלב 2: ניתוח מתקדם

- להשתמש ב־ Wireshark כדי לאתר שידורים חוזרים, חלונות אפס וכדומה.

שלב 3: פתרון פרקטי

- לאזן את התעבורה בין מסלולים (Load Balancing).
- לשדרג קישורים איטיים או עמוסים.
- להגדיל את התורים (Buffer Size) אם זה מתאים לחומרה.
- לכוון מחדש את פרוטוקולי הניתוב כדי להימנע מנתיבים בעייתיים.

במידת הצורך – לשנות את האלגוריתם של בקרת עומס בצד השרת/לקוח.

5) צעדים מומלצים לפתרון

איסוף נתונים: Packet Capture, חיפוש עומס RTT גבוה וניתוח לוגים.
בדיקת עקביות במסלול: Traceroute לזיהוי לולאות או קפיצות חריגות, השוואת הגדרות ניתוב בכל הרכיבים.
בדיקה של ממשקים פיזיים וחומרה: סקירת שגיאות, בדיקת כבלים.
וידוא הגדרות ICMP: בדיקות ping עם גדלים שונים, אימות, ווידוא שלא חוסמים ICMP.
כיוון או הגדרה מחדש של בקרת עומס ב-TCP: התאמת גדלתי חלונות ואלגוריתם הפעלת Window Scaling לרוחב פס גבוה.

לסיכום, אובדן חבילות גבוה בין שני נתבים עלול לנבוע משילוב של גורמים ב:
שכבת הרשת - כגון עומס, הגדרות ניתוב שגויות, או תקלות חומרה.
שכבת התעבורה - בעיקר כתוצאה מרגישות של פרוטוקולי TCP ו-UDP לאיבודים. על ידי בחינה שיטתית של הגדרות הרשת, הפרמטרים של שכבת התעבורה, וחיבורי החומרה, ניתן לאתר את מקור הבעיה וליישם פתרונות מתאימים לטיפול בה.

ניהול תהליך התחקור בשלבים:
איסוף לוגים, ניתוח תעבורה, התאמות הגדרות ובדיקות נקודתיות מוביל לזיהוי שורש הבעיה ולהטמעת פתרונות ארוכי טווח שימנעו תקלות חוזרות.

חלק 2

מאמרים

1. FlowPic: Encrypted Internet Traffic Classification is as Easy as Image Recognition.

2. Early Traffic Classification With Encrypted ClientHello: A Multi-Country Study.

3. Analyzing HTTPS Encrypted Traffic to Identify User's Operating System, Browser and Application.

FlowPic: Encrypted Internet Traffic Classification is as Easy as Image Recognition

1. התרומה העיקרית של המאמר:

תרומתו העיקרית של המאמר הינה הצגת גישה חדשנית לסיווג תעבורת אינטרנט מוצפנת המבוססת על המרת נתוני זרימה בסיסיים לתמונה המכונה FlowPic, ושימוש ברשתות נוירונים (CNN) לצורך ביצוע הסיווג, בדומה לבעיות של זיהוי תמונה. גישה זו מהווה שיפור משמעותי על פני שיטות קודמות שהסתמכו על מאפיינים המיוצרים באופן ידיני.

המאמר מראה כי שימוש ב FlowPics וב CNN מאפשר סיווג מדויק של קטגוריות תעבורה ואפליקציות, כולל תעבורה מוצפנת באמצעות VPN ו Tor מבלי להזדקק לבדיקת תוכן החבילות, ובכך לשמור על הפרטיות.

2. מאפייני התעבורה:

מאפיינים בסיסיים:

1. גדלי חבילות וזמני הגעה - מאפיינים אלו שימשו בעבר במחקרים לסיווג תעבורת רשת, לרוב כחלק מחילוץ תכונות סטטיסטיות. במאמר זה השימוש בהם נעשה בצורה חדשנית, כאשר הם ממופים לתמונה דו-ממדית (FlowPic) .
2. הגדרת זרם לפי 5 מאפיינים עיקריים (5-tuple) - המאמר משתמש בהגדרה של זרם רשת (פרוטוקול, כתובת IP מקור ויעד, פורט מקור ויעד) . הגדרה זו מאפשרת לפצל את התעבורה לקבוצות זרמים חד-כיווניים לצורך ניתוח מדויק.
3. זרמים קדמיים ואחוריים - המאמר מתייחס לתעבורה משני כיווני התקשורת (forward , backward) בשלב ההכנה של הנתונים, במטרה לבדוק שהזרם באמת שייך לפעולה הרצויה.

מאפיינים חדשניים:

1. סיווג תעבורה באמצעות המרת זרימת תעבורה לתמונה (PlowPic) – הגישה החדשנית היא הפיכת נתוני זרימה בסיסיים (גודל חבילה וזמן הגעת החבילה) לייצוג ויזואלי בצורת תמונה דו ממדית. כאשר ציר ה-X מיצג את זמן הגעת החבילה המנורמל וציר ה-Y מייצג את גודל החבילה. כל פיקסל בתמונה מייצג את מספר החבילות שהגיעו במרווח זמן ובגודל המתאימים.
2. באמצעות ניתוח CNN - במקום להפיק מאפיינים סטטיסטיים ידנית, המאמר מזין את תמונות ה FlowPic לרשת קונבולוציונית (CNN) שלומדת באופן אוטומטי את הדפוסים הרלוונטיים מתוך הייצוג הוויזואלי של זרם התעבורה, ומבצעת את הסיווג בדיוק גבוה וללא תלות בתהליך של בחירת מאפיינים ידנית.

3. תוצאות עיקריות

1. המחקר מציג שיטה חדשנית לסיווג תעבורת רשת באמצעות הפיכת זרמי תעבורה לתמונות (FlowPic), המודל הצליח לזהות את האפליקציה שממנה הגיע הזרם בדיוק כמעט מוחלט של 99.7%, בעוד ששיטות קודמות הגיעו לדיוק של 93.9%.

לצורך האימון, נעשה שימוש בחמישה סוגי שימוש נפוצים באינטרנט – שיחות קוליות, שיחות טקסט, העברת קבצים, צפייה בווידאו וגלישה באתרים.

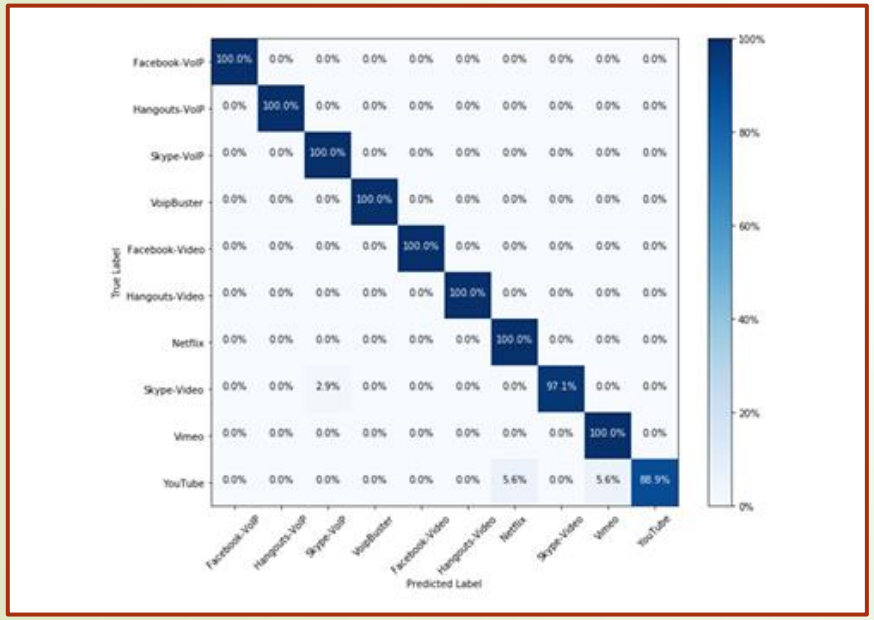
הבדיקות בוצעו בשלושה מצבי תעבורה: Non VPN , TOR , VPN.

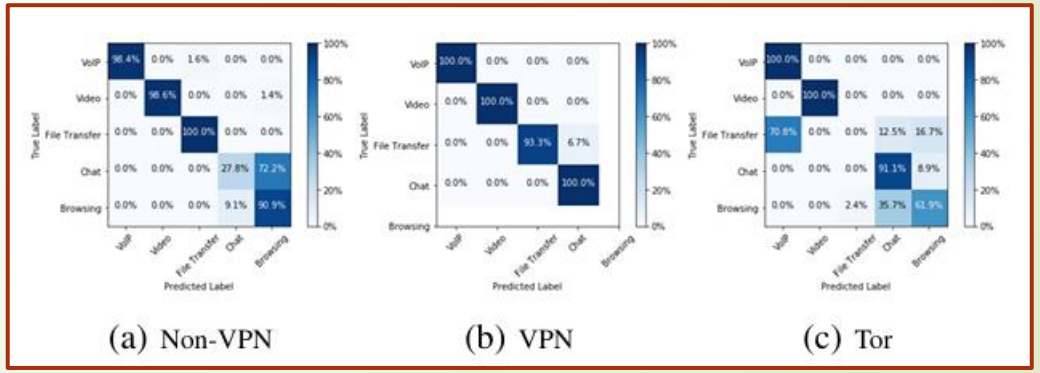
כאשר המודל התאמן על Non VPN הוא הצליח לזהות את סוג השימוש בדיוק של 96.2% בכל הקטגוריות, למעט גלישה, שזוהתה ב־90.6% בלבד.

בהתמודדות עם תעבורה מוצפנת מסוג VPN, רמת הדיוק עלתה ליותר מ־99.2%.

גם עם תעבורה דרך Tor, שנחשבת מאתגרת יותר, המודל הצליח לזהות את מרבית סוגי השימוש בדיוק של מעל 89% – למעט העברת קבצים, שבה נרשמה ירידה ל־55.8%.

תוצאות אלה ממחישות את העמידות של השיטה גם בתנאים של הצפנה חזקה, ואת היכולת שלה להכליל דפוסים בצורה טובה.





(a) Non-VPN (b) VPN (c) Tor

2. ממצא בולט מהמחקר הוא שהמודל הצליח לזהות סוגי תעבורה שהוא לא אומן עליהם כלומר, גם כשלא ראה תעבורה מסוימת במהלך האימון, הוא עדיין הצליח לזהות אותה ברמת דיוק מפתיעה.

למשל, כאשר המודל התאמן רק על תעבורה Non VPN הוא הצליח לזהות תעבורה שעברה דרך VPN בדיוק של כ 79% ותעבורה שעברה דרך Tor בדיוק ממוצע של 60%.

כמו כן, כשהמודל התאמן על VPN הוא הצליח לזהות תעבורת Non VPN ביותר מ 54%, ואת Tor ב54% .

כאשר המודל אומן רק על Tor הוא הצליח לזהות תעבורה Non VPN בדיוק של 52.1% ו- VPN ב- 35.8%.

Class	Accuracy (%)			
VoIP	Training/Test	Non-VPN	VPN	Tor
	Non-VPN	**99.6**	99.4	48.2
	VPN	95.8	**99.9**	58.1
	Tor	52.1	35.8	**93.3**
Video	Training/Test	Non-VPN	VPN	Tor
	Non-VPN	**99.9**	98.8	83.8
	VPN	54.0	**99.9**	57.8
	Tor	55.3	86.1	**99.9**
File Transfer	Training/Test	Non-VPN	VPN	Tor
	Non-VPN	**98.8**	79.9	60.6
	VPN	65.1	**99.9**	54.5
	Tor	63.1	35.8	**55.8**
Chat	Training/Test	Non-VPN	VPN	Tor
	Non-VPN	**96.2**	78.9	70.3
	VPN	71.7	**99.2**	69.4
	Tor	85.8	93.1	**89.0**
Browsing	Training/Test	Non-VPN	VPN	Tor
	Non-VPN	**90.6**	-	57.2
	VPN	-	-	-
	Tor	76.1	-	**90.6**

3. מסקנות עיקריות

1. גישת FlowPic יחד עם רשתות עצביות קונבולוציוניות CNNs משיגה דיוק גבוה מאוד בסיווג תעבורת אינטרנט מוצפנת.
2. התוצאות במאמר מראות כי סיווג קטגוריות תעבורת אינטרנט דרך Tor הוא מאתגר יותר בהשוואה לסיווג על פני VPN ו-Non-VPN.
3. היתרון המרכזי של גישת FlowPic הוא הגנריות וחוסר התלות ביישום ספציפי . בשונה משיטות מסורתיות שמתאימות את עצמן לדפוסי תעבורה של אפליקציות מסוימות. FlowPic מתמקדת במאפייני ההתנהגות הכלליים גודל ותזמון החבילות ולכן מסוגלת לזהות גם אפליקציות חדשות שלא נראו בשלבי האימון.
4. יתרון נוסף שמציין המאמר הוא שהגישה אינה מסתמכת על תוכן המטען (payload) של החבילות ,ובכך אינה פוגעת בפרטיות. דרישת האחסון שלה מינימלית.

Early Traffic Classification With Encrypted ClientHello: A Multi-Country Study

1. התרומה העיקרית של המאמר:

התרומה המרכזית של המאמר היא פיתוח וניתוח של אלגוריתם חדש לסיווג מוקדם של תעבורת רשת מוצפנת בתרחיש של Encrypted ClientHello שנקרא hRFTC (hybrid Random Forest Traffic Classifier).

hRFTC מייצג התקדמות משמעותית בתחום סיווג התעבורה המוצפנת בכך שהוא משלב באופן היברידי מאפיינים לא מוצפנים של פרוטוקול TLS עם מאפיינים סטטיסטיים של זרימת תעבורה. גישה זו מאפשרת לו לנצל מידע זמין בשלבים הראשונים של החיבור, לפני הגעת נתוני האפליקציה המוצפנים.

תוצאות המחקר מראות כי אלגוריתמים המסתכמים על מאפייני TLS משיגים 38.4% זיהוי תעבורה בעוד כי הגישה ההיברידית של hRFTC משפרת באופן דרמטי את יעילות סיווג התעבורה ומגיעה ל 94.6% זיהוי.

2. מאפייני התעבורה:

מאפיינים בסיסיים:

1. מאפייני מבוססי מנות:
מאפיינים המתמקדים במידע הלא מוצפן בהודעות TLS handshake , הכוללים פרטים כמו גרסת TLS, key share - רשימת שיטות הצפנה שהלקוח תומך בהן, Key Share Groups - פרמטרים המגדירים את סוג מפתח ההצפנה , extensions אורך רשימת ההרחבות, גודל ההודעות.

2. מאפיינים מבוססי זרימה:
מאפיינים אלה נגזרים מרצפים של גדלי מנות וזמני ההגעה בין מנות. המאפיינים מחושבים בנפרד עבור מנות עולות (uplink - UL) ויורדות (downlink- DL).
סט המאפיינים מבוסס הזרימה כולל סטטיסטיקות שונות הכוללות : ספירה, ממוצע, סטיית תקן, ערך מינימלי ומקסימלי ואחוזונים, בנוסף גם סטטיסטיקת גודל החבילות.

מאפיינים חדשניים:

המאמר מצביע על מספר היבטים חדשניים בשימוש במאפיינים על ידי hRFTC:

1. הגישה ההיברידית - שילוב של מאפיינים מבוססי מנות עם מאפיינים מבוססי זרימה במטרה לבצע סיווג מוקדם של תעבורה.
2. קריטריון חדש לבחירת חבילות - במקום לקחת מספר קבוע של חבילות, האלגוריתם מחכה רק עד החבילה הראשונה עם נתוני אפליקציה מהשרת ,וכך משיג איחור מזערי אבל מידע מקסימלי.
3. התאמה ל־ QUIC מבנה ההודעות ב- QUIC שונה מזה של TLS רגיל. האלגוריתם hRFTC מותאם למבנה הזה וממפה את השדות החדשים כמו (QUIC version ו-Connection ID) לתוך וקטור המאפיינים.

3. תוצאות עיקריות

1. ביצועים גבוהים של hRFTC לעומת אלגוריתמים קיימים:

במרכז המחקר ניצב האלגוריתם החדש hRFTC , שנועד לסווג תעבורת רשת מוצפנת בצורה יותר מדויקת, גם כאשר פרוטוקול Encrypted ClientHello מסתיר מידע חיוני.

האלגוריתם משלב מאפיינים מתוך הודעות TLS לא מוצפנות יחד עם תכונות סטטיסטיות של זרימות התעבורה. בניסויים שנערכו hRFTCהגיע לדיוק סיווג של עד 94.6%, לעומת 38.4% בלבד באלגוריתמים מבוססי חבילה , מה שמעיד על יתרון משמעותי ביכולת לזהות סוגי שירותים שונים למרות ההצפנה. באלגוריתמים מבוססי זרימה אמנם הגיעו לתוצאות טובות אך לא כמו אלגוריתם hRFTC.

Class	F-score [%]						
	Hybrid Classifiers			Flow-based Classifier	Packet-based Classifiers		
	hRFTC [proposed]	**UW** [35]	**hC4.5** [34]	**CESNET** [63]	**RB-RF** [24]	**MATEC** [33]	**BGRUA** [32]
BA-AppleMusic	**92.1**	89.5	80.2	89.2	25.5	13.1	14.5
BA-SoundCloud	**99.6**	98.9	97.8	98.7	84.4	81.8	82.0
BA-Spotify	**93.6**	90.8	89.0	88.5	16.3	0.0	3.6
BA-VkMusic	**95.7**	89.7	88.5	91.8	2.6	2.1	3.2
BA-YandexMusic	**98.5**	93.2	93.7	92.5	1.8	0.2	0.1
LV-Facebook	**100.0**	99.7	99.8	99.8	**100.0**	**100.0**	**100.0**
LV-YouTube	**100.0**	**100.0**	99.9	**100.0**	**100.0**	99.0	98.4
SBV-Instagram	**89.7**	74.7	76.5	78.8	10.0	6.3	6.4
SBV-TikTok	**93.3**	81.8	81.8	76.3	38.3	34.3	34.5
SBV-VkClips	**95.7**	94.0	91.3	92.4	53.2	37.7	46.0
SBV-YouTube	**98.2**	96.6	94.7	96.4	1.1	0.2	0.2
BV-Facebook	**87.7**	78.2	79.7	77.6	5.6	3.2	3.8
BV-Kinopoisk	**94.1**	84.1	85.8	89.8	5.4	4.0	4.1
BV-Netflix	**98.5**	97.2	95.2	93.7	50.7	52.3	56.1
BV-PrimeVideo	**91.3**	86.7	84.1	84.7	32.5	24.7	26.8
BV-Vimeo	**94.8**	90.5	90.2	81.4	72.0	19.5	68.6
BV-VkVideo	**88.6**	80.5	80.4	79.7	10.5	0.0	0.1
BV-YouTube	**85.9**	84.3	77.0	78.5	22.3	19.6	20.2
Web (known)	**99.7**	99.5	99.4	99.4	98.0	98.0	98.0
Macro-F-score (average)	**94.6**	89.9	88.7	88.9	38.4	31.4	35.1

2. ביצועי סיווג התעבורה המוקדם תלויים במיקום הגיאוגרפי:

אלגוריתמים שאומנו על תעבורה ממדינה אחת מתקשים לסווג במדויק תעבורה ממדינה אחרת, המשמעות היא שמאפייני התעבורה, אפילו אלה שאינם קשורים ישירות לתוכן האפליקציה (כמו גודל מנות וזמני הגעה), יכולים להשתנות בין מיקומים גיאוגרפיים שונים, לכן כדי להשיג דיוק גבוה יש צורך לאמן מודלים של סיווג תעבורה בנפרד עבור כל מיקום גיאוגרפי שבו הם אמורים לפעול.

Test Country	Share in Dataset	Training Country	Classifier Macro F-score [%]		
			hRFTC	hC4.5	UW
Germany	18.8%	Others	38.4	26.9	19.5
Kazakhstan	3.0%	Others	57.3	32.3	27.5
Russia	29.2%	Others	49.8	35.6	20.9
Spain	16.3%	Others	38.5	34.4	12.6
Turkey	25.2%	Others	35.1	26.0	16.4
USA	7.5%	Others	49.2	41.4	21.3

3. קושי בזיהוי תעבורה עם QUIC

כאשר נעשה שימוש בפרוטוקול QUIC יחד עם Padding כל החבילות מתקבלות בגודל דומה, מה שמטשטש את ההבדלים בין סוגי שירותים. כתוצאה מכך, אלגוריתמים דינמיים מתקשים לסווג את התעבורה בדיוק גבוה.

3. מסקנות עיקריות

1. אלגוריתמים שמסתמכים רק על מידע מתוך לחיצת היד TLS לא מצליחים לשמור על דיוק כאשר נעשה שימוש בפרוטוקול ECH שמצפין את המידע.
2. ישנם הבדלים גאוגרפיים ברשת , כמו ספקי CDN, עיכובים, גודל חבילה ותשתיות שונות. יש צורך לאמן את המודל באופן מקומי בהתאם לאזור שבו הוא פועל.
3. דיוק גבוה של אלגוריתם hRFTC גם עם מעט נתוני אימון, גם כאשר מאמנים את האלגוריתם על 10% מהמאגר הדיוק נשאר גבוה. זה מראה כי השיטה לא דורשת כמויות גדולות של מידע כדי לתפקד היטב.
4. המאמר מציג כי גודל החבילה הוא המאפיין העיקרי התורם את החלק הגדול ביותר לדיוק הסיווג –כ-50%, בנוסף מאפייני TLS שלא הוצפנו (כמו מספר ההרחבות, אורך ההודעה, קבוצות הצפנה) עדיין תורמים כ30% נוספים לדיוק.
5. פרוטוקול QUIC עם Padding מקשה על סיווג מדויק אך עדיין מאפשר אותו ברמה סבירה.

Analyzing HTTPS Encrypted Traffic to Identify User's Operating System, Browser and Application

1. התרומה העיקרית של המאמר:

החוקרים מציגים תרומה משמעותית לתחום ניתוח תעבורת הרשת והפרטיות. התרומה המרכזית של המאמר היא ההדגמה של היכולת לזהות באופן מדויק את מערכת ההפעלה, הדפדפן והיישום של המשתמש על סמך ניתוח פסיבי של תעבורת HTTPS מוצפנת בלבד באמצעות ניתוח דפוסי תעבורה ולמידת מכונה.

החוקרים פיתחו תכונות חדשות לזיהוי הכוללות התנהגות "מתפרצת" (bursty) של תעבורת דפדפן ומאפיינים של פרוטוקול ההצפנה TLS/SSL.

בנוסף, כחלק מתרומתם, החוקרים אספו ויצרו מאגר נתונים מקיף של למעלה מ20,000 דגימות של תעבורת HTTPS מוצפנת, שכל דוגמה מתויגת לפי מערכת ההפעלה, הדפדפן והיישום. המאגר כולל תעבורה מ Windows, OSX ו Ubuntu, דפדפנים כמו Chrome, Firefox, Safari ואפליקציות כמו YouTube, Facebook ו־Twitter.

2. מאפייני התעבורה:

מאפיינים בסיסיים:

Forward packets

Forward total Bytes

Min forward inter arrival time difference

Max forward inter arrival time difference

Mean forward inter arrival time difference

STD forward inter arrival time difference

Mean forward packets

STD forward packets

Backward packets

Backward total Bytes

Min backward inter arrival time difference

Max backward inter arrival time difference

Mean backward inter arrival time difference

STD backward inter arrival time difference

Mean backward packets

STD backward packets

Mean forward TTL value

Minimum forward packet

Minimum backward packet

Maximum forward packet

Maximum backward packet

Total packets #

Minimum packet size

Maximum packet size

Mean packet size

Packet size variance

מאפיינים חדשניים:

Forward min peak throughput

Mean throughput of forward peaks

Forward STD peak throughput

Mean backward peak inter arrival time diff

Minimum backward peak inter arrival time diff

Maximum backward peak inter arrival time diff

STD backward peak inter arrival time diff

Mean forward peak inter arrival time diff

Minimum forward peak inter arrival time diff

Maximum forward peak inter arrival time diff

STD forward peak inter arrival time diff

Keep alive packets

TCP Maxiumu Segment Size

Forward SSL Version

TCP window scaling factor

SSL compression methods

SSL extension count

SSL chiper methods

SSL session ID len

Forward peak MAX throughput

Mean throughput of backward peaks

Max throughput of backward peaks

Backward min peak throughput

Backward STD peak throughput

Forward number of bursts

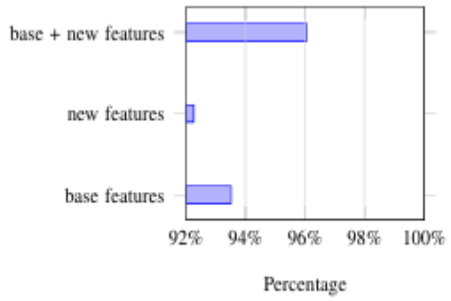
Backward number of bursts

3. תוצאות עיקריות

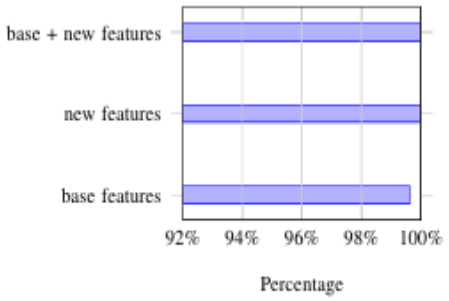
התוצאה העיקרית של המאמר מראה כי ניתן לבצע סיווג בדיוק גבוה של מערכת ההפעלה, הדפדפן והיישום של משתמשים על ידי ניתוח תעבורת HTTPS מוצפנת, תוך שימוש במאפייני תעבורה בסיסיים וחדשים.

איור 2 במאמר מציג את תוצאות הדיוק, ננתח את התוצאות:

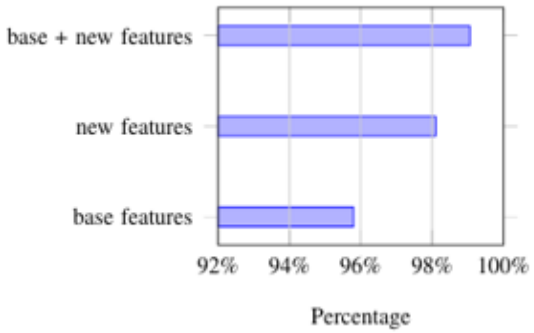
1. ניתן לשים לב כי תוצאות דיוק שלושת האלמנטים על ידי שילוב מאפייני הבסיס והמאפיינים החדשים עלה משמעותית (96%) ביחס לדיוק המאפיינים לבדם.
2. התוצאות עבור מערכת ההפעלה מראות כי דיוק הסיווג היה גבוה מאוד בכל שלושת קבוצות המאפיינים, ניתן לראות כי קל לזהות את מערכת ההפעלה גם לפי המאפיינים הבסיסיים בלבד.
3. תוצאות דיוק הדפדפן - דיוק סיווג הדפדפן הינו גבוה, כאשר השילוב של מאפייני הבסיס והחדשים הניב את התוצאה הטובה ביותר (99%). המאפיינים החדשים העלו את רמת דיוק זיהוי הדפדפן.
4. תוצאות דיוק היישום – גם כאן השילוב של מאפייני הבסיס והחדשים הניב את הדיוק הגבוה ביותר, המאפיינים החדשים הביאו לשיפור הדיוק, מעל 96%.

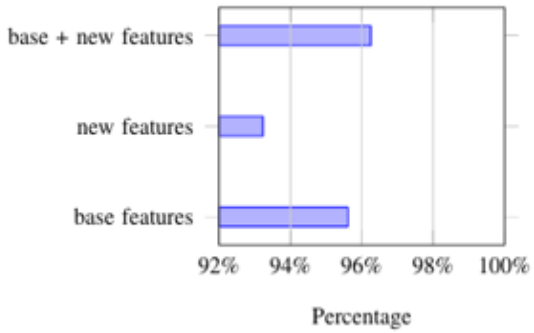


(a) Tuple Accuracy Results



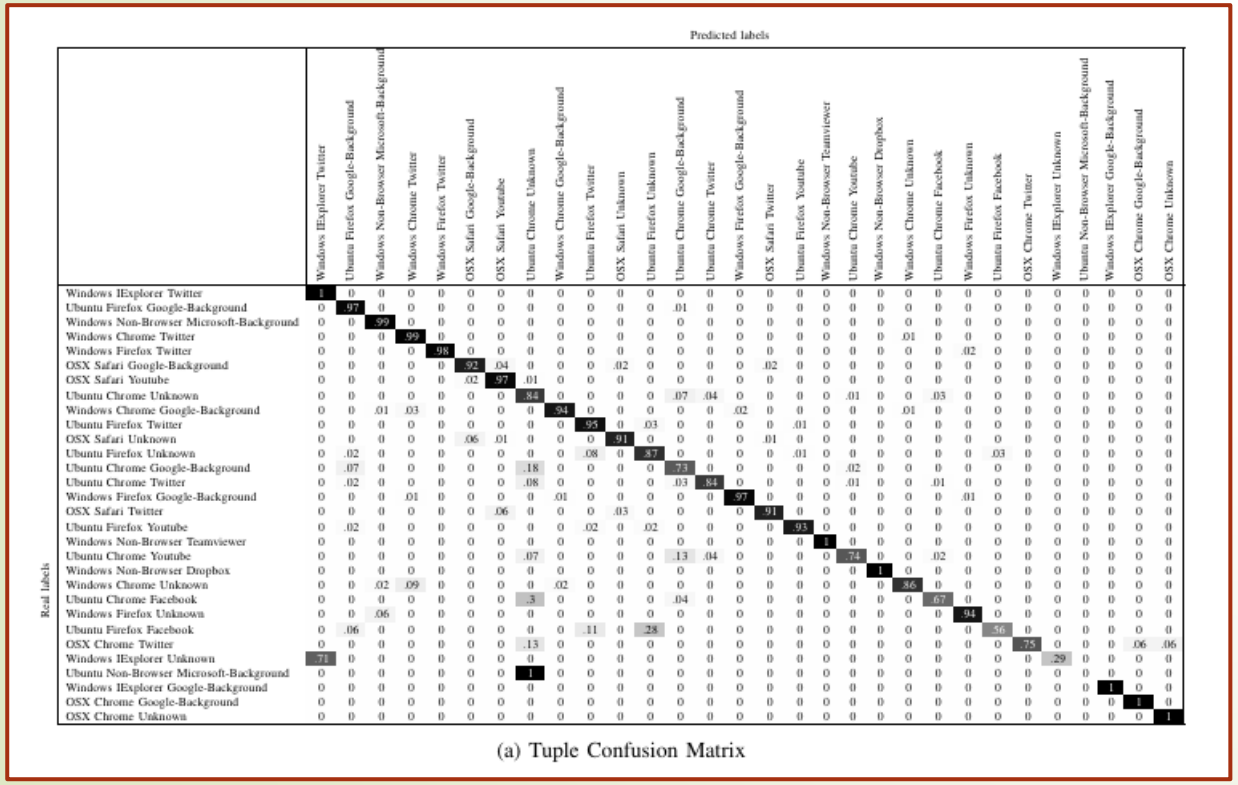
(b) OS Accuracy Results



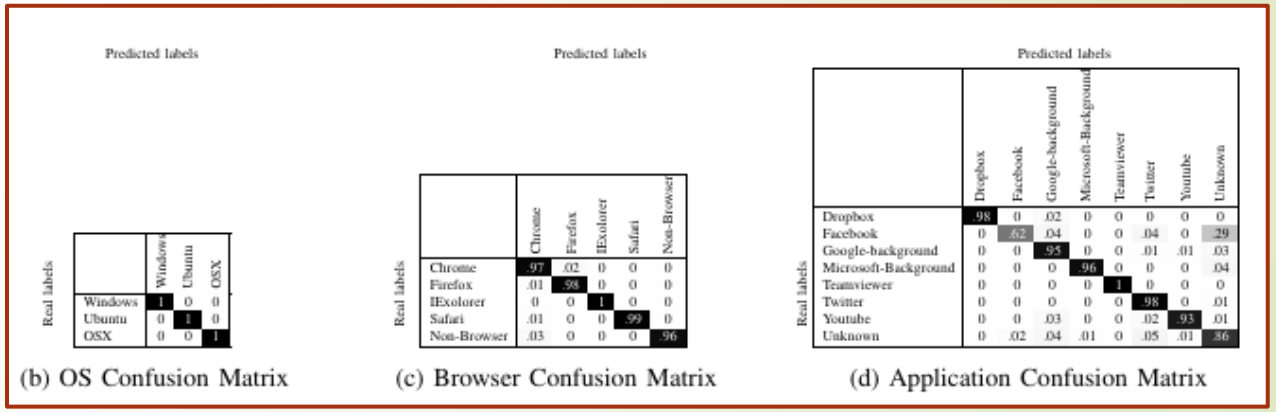


איור 4 מציג מטריצת בלבול (Confusion Matrices) אשר מתארת את סוגי הטעויות שנעשו על ידי המודל בזיהוי האלמנטים.

1. סיווג שלושת המאפיינים סווגו בדיוק גבוה לרוב, מטריצת הבלבול של השלושה מראה כי רוב השילובים של מערכת הפעלה, דפדפן ויישום סווגו נכון באחוזים גבוהים, כפי שניתן לראות מערכי האלכסון הראשי של המטריצה.
2. זיהוי מערכת ההפעלה הינו מושלם Windows, Ubuntu ו OSXסווגו נכון תמיד.
3. זיהוי הדפדפן כמעט מדויק ,עם מעט טעויות בין דפדפנים דומים למשל Chrome ו Firefox.
4. מרבית הטעויות בסיווג היישום נובעות בעיקר מסיווג שגוי לקטגוריות דומות או לקטגוריה "לא ידוע".



(a) Tuple Confusion Matrix



(b) OS Confusion Matrix (c) Browser Confusion Matrix (d) Application Confusion Matrix

3. מסקנות עיקריות

1. ניתן לזהות משתמשים בדיוק גבוה מאוד. המודל שפיתחו החוקרים הצליח לזהות שלושה מאפיינים : מערכת הפעלה, דפדפן ויישום בדיוק של 96.06%.
2. מערכת הפעלה הינה קלה יותר לזיהוי מיישומים ודפדפנים , מתוך הממצאים עולה כי זיהוי מערכת ההפעלה היה הקל ביותר לזיהוי, אחר כך דפדפן ולאחר מכן יישום (לפי רמת קושי הזיהוי ודיוקו).
3. התנהגות מתפרצת של דפדפנים היא מפתח לזיהוי, מקטעים קצרים של שידור אינטנסיבי עם שקט לפני ואחרי מהווים חתימה ייחודית לדפדפן וליישום.
4. סיכונים לפגיעה בפרטיות, התובנות מהמחקר מדגישות את היכולת של תוקף פסיבי להסיק מסקנות משמעותיות על משתמשים.
5. הצפנה לבדה אינה מספיקה כדי להבטיח פרטיות, גם כאשר התעבורה מוצפנת (HTTPS), ניתן לגלות מידע פרטי על משתמש.

חלק 3

הסבר על ההקלטות

1. **Chrome** ו-**Firefox** (דפדפנים) – בהקלטה של כל אחד מהם, פתחנו את האפליקציה והגענו לדף הבית (Google), חיפשנו במנוע החיפוש "אוניברסיטת אריאל" ונכנסנו לאתר האוניברסיטה.

2. **Spotify** – בהקלטה נכנסנו לדף האינטרנט של Spotify דרך דפדפן Chrome והפעלנו פודקאסט (ללא וידאו).

3. **YouTube** – בהקלטה נכנסנו לדף האינטרנט של YouTube דרך דפדפן Chrome והפעלנו סרטון (אשר מורכב מווידאו ושמע).

4. **Zoom** – בהקלטה נכנסנו לאפליקציית Zoom במחשב וביצענו שיחת זום בין שני מחשבים שונים, הכוללת שימוש במצלמה, במיקרופון ובצ'אט.

לפני שהתחלנו את ההקלטות ומכיוון שקיים שימוש ב-TLS, דאגנו לשמור קובץ מפתחות והשתמשנו בו לצורך פענוח התעבורה. כמו כן, במהלך ההקלטות הדפדפנים עשו שימוש ב-QUIC (אפשרנו את האופציה).

סינונים שביצענו להקלטות

1. **Chrome** ו-**Firefox**:
- שליפת כתובות IP מתעבורת HTTP - באמצעות סינון על "http.host" אוספים את כל כתובות ה-IP של השרתים שאליהם נשלחו בקשות HTTP.
- שליפת כתובות IP מתעבורת TLS - באמצעות סינון על "tls.handshake.extensions_server_name" אוספים את כל כתובות ה-IP של השרתים שקשורים לתעבורת ה-TLS.

2. **Spotify**:

```
tcp.port == 4070 || tcp.port == 443 || udp.port == 443
```

- פורט 4070 הוא פורט ייעודי בו Spotify משתמשת על מנת להעביר מוזיקה כשהיא לא משתמשת ב-https רגיל.
- פורט 443 – נתייחס לשני הפרוטוקולים הרלוונטיים בסינון על פי פורט זה:

UDP - מכיוון שהדפדפן Chrome, שדרכו הפעלנו את Spotify, תומך ב-QUIC (כפי שציינו בשקופית הקודמת), רוב הבקשות ל-Spotify דרך הדפדפן עברו דרך UDP במקום TCP.

TCP - אם חלק מהתעבורה לא השתמשה ב-QUIC, היא הייתה מתבצעת על גבי TLS (HTTPS) בפורט זה.

סינונים שביצענו להקלטות

3. YouTube:

```
tcp.port == 443 || ip.dst == 142.250.0.0/16
```

- טווח הכתובות 142.250.0.0/16 שייך לגוגל ובפרט לשרתי YouTube. זה מסנן את התעבורה מול YouTube ומסיר רעשים מתעבורה מול אתרים אחרים.
- פורט 443 מסנן את כל תעבורת ה- HTTPS של YouTube על גבי פרוטוקול TCP.

4. Zoom:

```
(udp.port >= 8801 && udp.port <= 8810) || tcp.port == 443
```

- Zoom משתמשת בטווח הפורטים הזה עבור שידור אודיו ווידאו בזמן אמת בפרוטוקול UDP.
- פורט 443 כלול כי Zoom משתמשת לעיתים גם ב-TCP מוצפן.

הסבר על הסקריפטים

- **plot_network_traffic.py**

הסקריפט מנתח את ההקלטות שביצענו (קובץ pcap), בכך שהוא מחלץ מידע על תעבורת הרשת ומציג אותו בצורה גרפית.
הסקריפט מבצע שלושה סוגי ניתוחים על בסיס (פירוט בהמשך המצגת):

1. IP header fields
2. TCP header fields
3. TLS header fields

- **plot_packet_sizes.py**

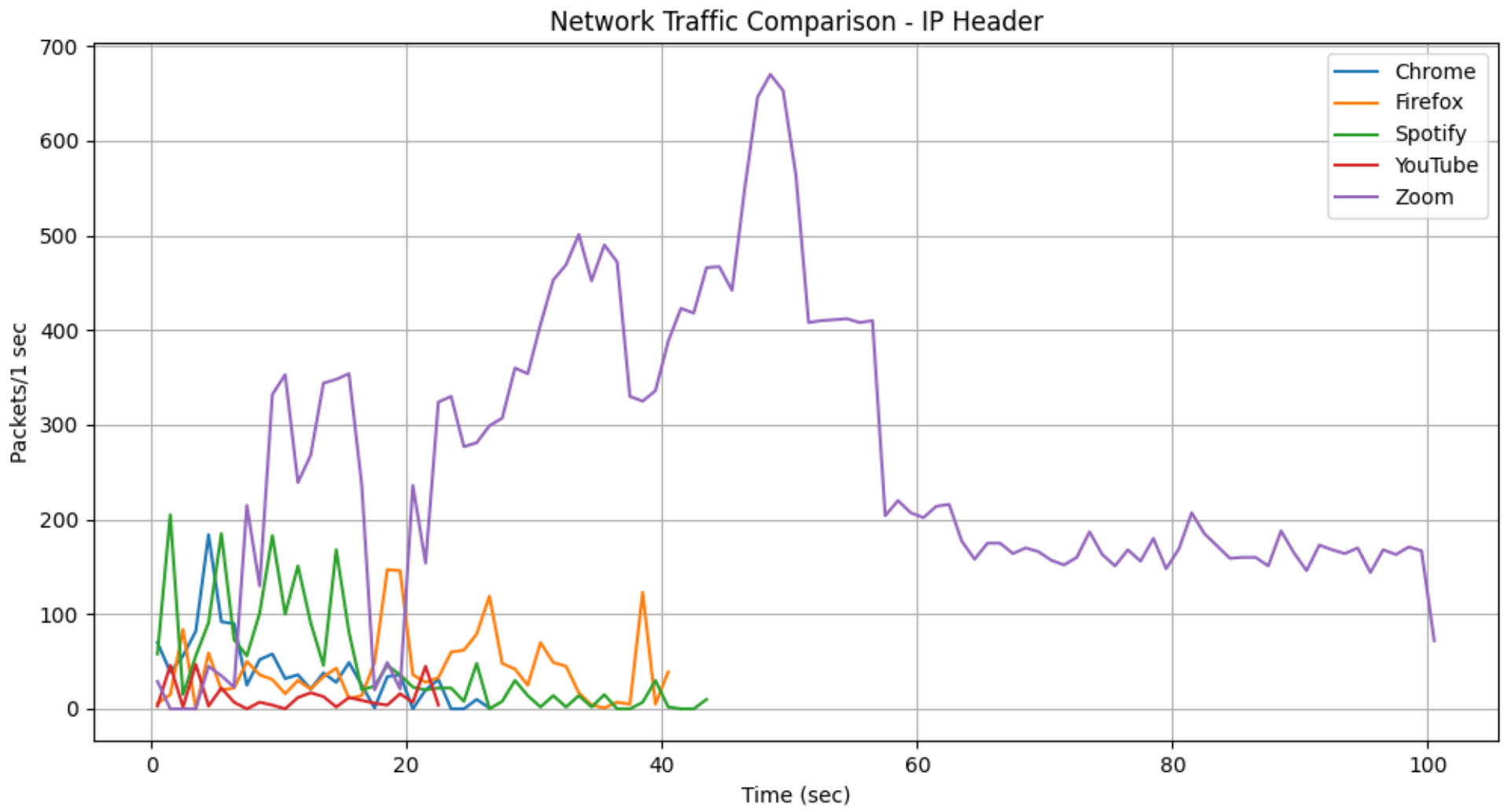
הסקריפט מנתח את ההקלטות שביצענו (קובץ pcap), בכך שהוא מחלץ מידע על תעבורת הרשת ומציג אותו בצורה גרפית.
הסקריפט מבצע ניתוח ייחודי, על בסיס גודל החבילות ויוצר גרפים עבור (פירוט בהמשך המצגת):

1. חבילות קטנות (חבילות שגודלן קטן מ-200 בתים)
2. חבילות בינוניות (חבילות שגודלן בין 200 בתים ל-1000 בתים)
3. חבילות גדולות (חבילות שגודלן מעל ל-1000 בתים)

שני הסקריפטים משתמשים בספריית pyshark לקריאת קבצי ההקלטה וב-matplotlib ליצירת הגרפים.

כדי להריץ את הסקריפטים, יש לוודא כי מזינים את הניתוב ושמות הקבצים של ההקלטות תחת apps המוגדר בראש כל סקריפט.

IP header fields – גרף א'



גרף א' – IP header fields

הקדמה:
בגרף זה ניתן להבחין בהבדלים בין האפליקציות, כך שעבור כל האפליקציה ניתן לראות את אופן ניצול שכבת ה-IP. בשפה פשוטה, הגרף מראה איזו אפליקציה שולחת הכי הרבה נתונים וכמה "עמוסה" התעבורה שלה. אם אפליקציה שולחת הרבה חבילות IP, זה אומר שהיא מתקשרת בתדירות גבוהה עם השרתים שלה.

צירי הגרף:
- **ציר ה-X**: מציין את הזמן לפי שניות.
- **ציר ה-Y**: מציין את כמות החבילות לאינטרוול (פרק זמן של שנייה אחת).

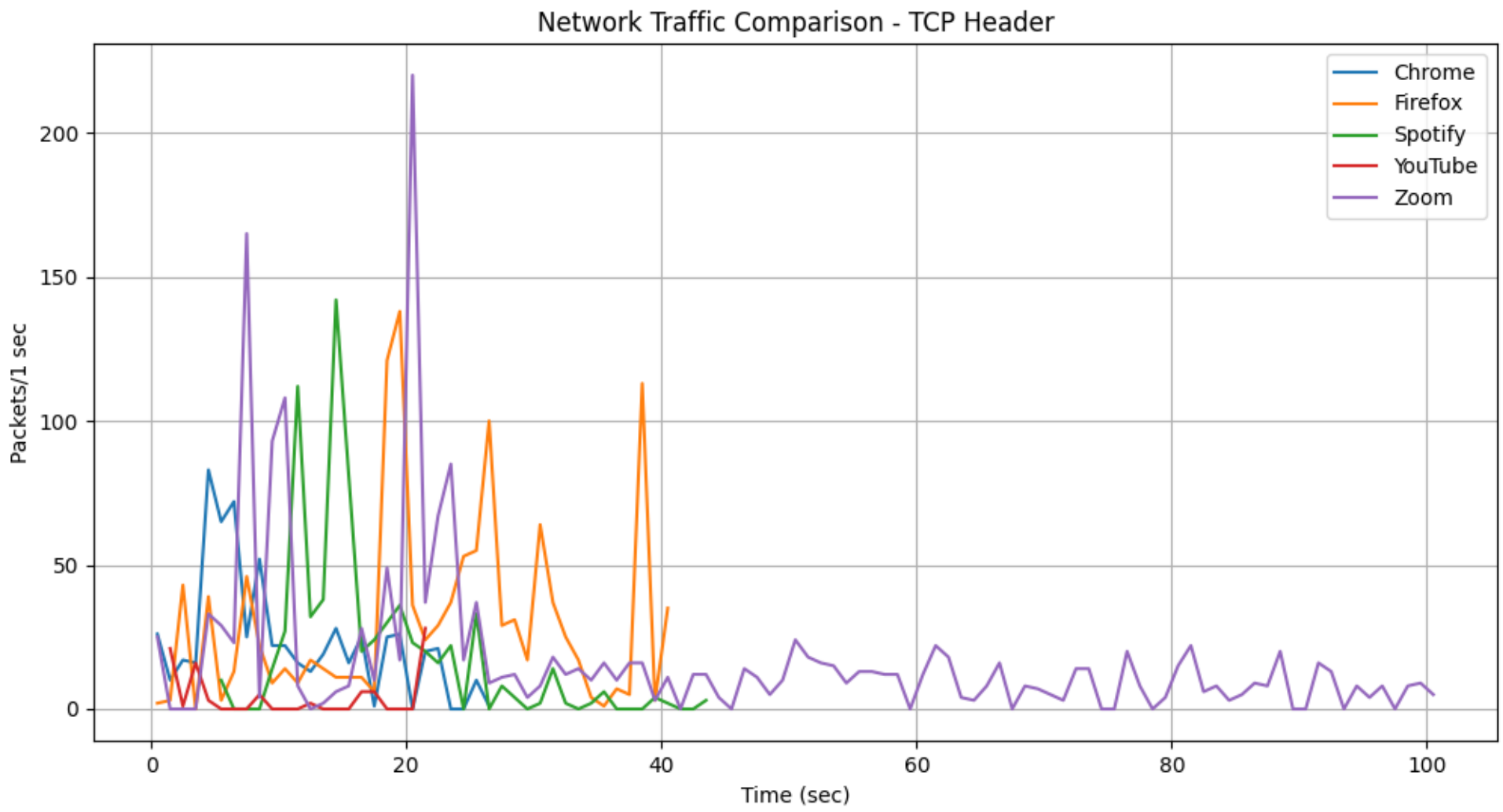
השוואה:
1. **Chrome** ו-**Firefox** (דפדפנים) – ניתן לראות שאין דפוס קבוע של תנודות בגרף. עם זאת, ניתן להבחין כי כאשר נטען דף אינטרנט חדש, גדלה כמות החבילות לשנייה, מכיוון שיש שליחת נתונים (למשל טעינת תמונות).
2. **Spotify** – כפי שניתן לראות בגרף, בתחילת ההקלטה, כמות החבילות לכל שנייה הייתה יחסית גדולה, שכן התבצעה שליחת נתונים רבים הקשורים לבחירת תוכן השמע. לאחר שבחרנו את השמע והפעלנו אותו, אנחנו רואים כי בגרף ממשיך להתקיים מצב של הרבה חבילות לפרק זמן. לאחר מכן, הגרף מתאזן ומגיע לרמה כמעט אפסית של חבילות, כלומר תחילה היו הרבה חבילות שנשלחו למען דפדוף ובחירת התוכן המבוקש, לאחר שנבחר האודיו אותו אנו בחרנו להפעיל נשלחו חבילות רבות לשנייה לצורך טעינת הקובץ, ולאחר שנבחר הקובץ כמות החבילות לשנייה יורדת בצורה משמעותית . זה נובע מאופן ההזרמה הקבוע של השמע. כלומר, תחילה נשלחות הרבה חבילות לפרק זמן לצורך טעינה, בין אם של עמודי האפליקציה ובין אם של טעינת קטע האודיו, אך מתייצב לאחר שהאודיו נטען.

גרף א' – IP header fields

השוואה:

3. **YouTube** – באופן דומה למתואר תחת אפליקציית Spotify (ראה סעיף 2), בתחילת ההקלטה נטענו אובייקטים שונים (ביניהם תמונות וסרטונים בדף הפתיחה של יוטיוב) אך לאחר שנבחר סרטון רצוי ניתן לראות שהתעבורה הפכה לפחות תנודתית וכמות החבילות לשנייה ירדה. כמו כן, לקראת סוף ההקלטה, קפצה פרסומת, מה שמסביר את העלייה של הגרף בסוף.
4. **Zoom** – קל לראות בגרף, שנשלחות הרבה חבילות ביחס לאפליקציות האחרות. הסיבה לכך היא ששיחות וידאו דורשות שליחת נתונים בזמן אמת. הנתונים נשלחים באופן שאינו סדיר, בעקבות גורמים שונים.

TCP header fields – גרף ב'



גרף ב' – TCP header fields

הקדמה:
בגרף זה ניתן להבחין בהבדלים בין האפליקציות, כך שעבור כל האפליקציה ניתן לראות את אופן ניצול שכבת ה-TCP. בשפה פשוטה, הגרף מציג את מספר המנות הנשלחות בכל שנייה עבור 5 האפליקציות השונות, בין המחשבים לשרת. נוכל לראות בו כמה חיבורים נפתחים וכמה תעבורה TCP נוצרת לאורך זמן.

צירי הגרף:
- **ציר ה-X**: מציין את הזמן לפי שניות.
- **ציר ה-Y**: מציין את כמות החבילות לאינטרוול (פרק זמן של שנייה אחת).

השוואה:
1. **Chrome** ו-**Firefox** (דפדפנים) – בדומה לגרף הקודם ניתן לראות גם כאן שאין דפוס קבוע. כלומר, ניתן לראות שכמות החבילות לשנייה עולה כאשר אנו נכנסים לאתר מסוים. הסיבה לכך היא, כאשר אנו נכנסים לאתר מסוים מתבצע חיבור עם השרת. כלומר, מתבצע TCP 3-Way Handshake בין הלקוח לשרת, ולאחר מכן מתחילה תקשורת של SYN/ACK, שכפי שנראה בגרף, צורכת פחות העברת חבילות בשנייה.

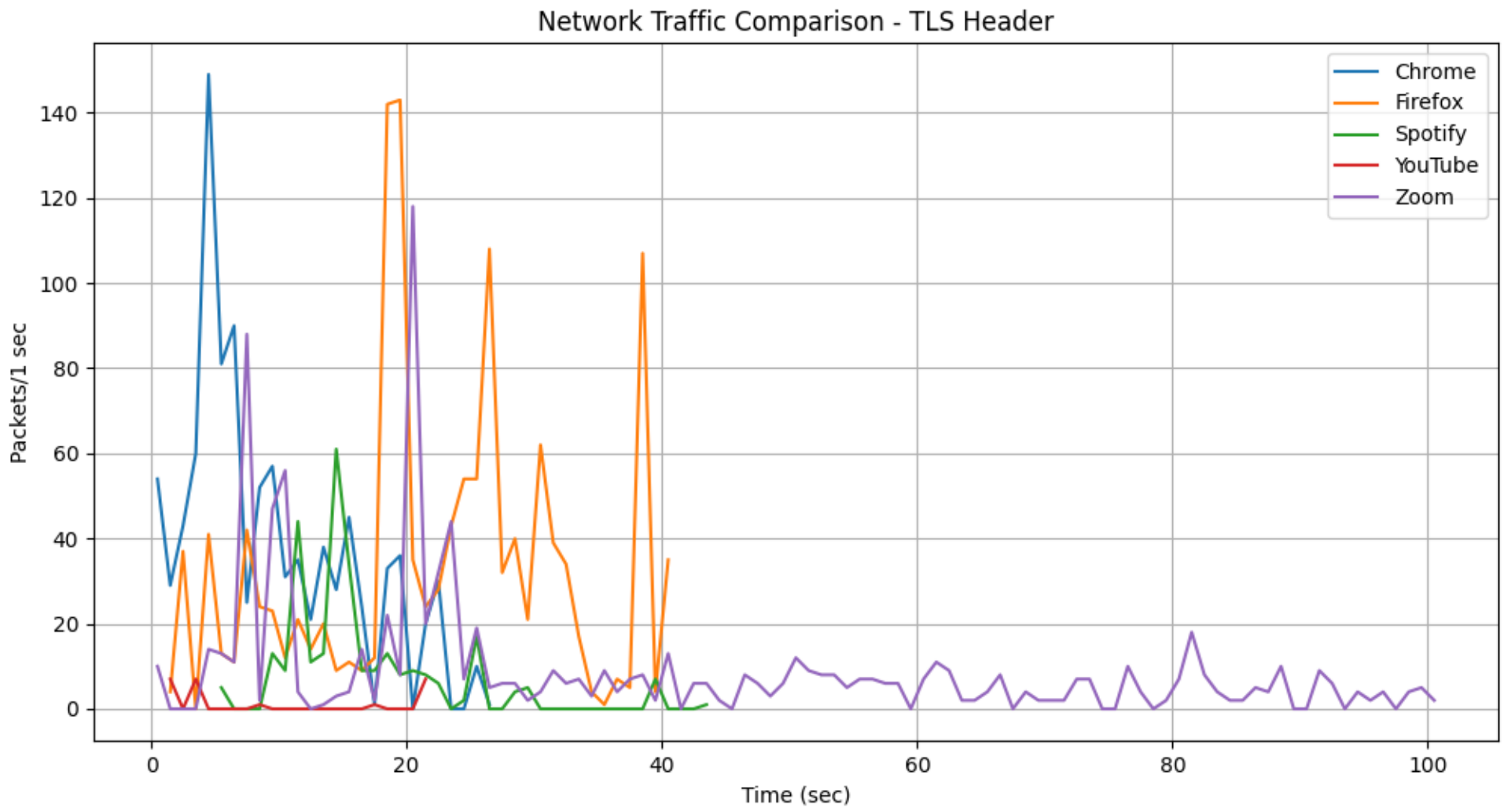
2. **Spotify** - כפי שניתן לראות בגרף, עבור Spotify תחילה מתבצע חיבור לשרת הכולל מנגנון TCP 3-way handshake, שמתבטא בכמות גבוהה של חבילות לפרק זמן של שנייה. לאחר מכן, כאשר מתחיל הניגון והוא כבר פעיל ויציב, התקשורת ממשיכה באופן רציף אך מתון. ניתן לפרש זאת בכך שהזרמת אודיו (כפי שגם ראינו בגרף הקודם) דורשת שליחה של חבילות קטנות בקצב קבוע. ניתן לראות זאת גם בגרף זה, שכן לאחר שהאפליקציה מתייצבת והקטע כבר נטען אז כמות החבילות לפרק זמן של שניה קטנה באופן משמעותי.

גרף ב' – TCP header fields

3. **YouTube** - עבור אפליקציה זו, ניתן לראות שעיקר התקשורת בין המחשב לשרת התבצע כאשר גללנו באתר והפעלנו את הסרטון. ניתן לראות שכאשר הסרטון כבר פעל העברת התקשורת הייתה מינימלית עד כדי אפסית. עם זאת, ניתן לראות שכאשר הופיעה פרסומת, התבצעה תקשורת רבה בין הלקוח לבין השרת.

4. **Zoom** - כפי שהסברנו עבור הגרף הקודם, עבור Zoom מתבצעת תקשורת רציפה. כלומר, בתחילת השיחה הכוללת התקשרות בין שני המחשבים וייצוב הקשר נשלחות הרבה חבילות בין המחשב לשרת. אך כאשר החיבור כבר מתייצב, ניתן לראות שהחיבור הוא קבוע באופן יחסי.

TLS header fields – גרף ג'



גרף ג' – TLS header fields

הקדמה:
בגרף זה ניתן להבחין בהבדלים בין האפליקציות, כך שעבור כל האפליקציה ניתן לראות את אופן ניצול שכבת ה-TLS. נזכיר כי TLS משמש להצפנת מידע באפליקציות. בשפה פשוטה, הגרף מראה איזו מהאפליקציות משתמשת בתקשורת מוצפנת ואיך זה משתנה לאורך זמן. אנחנו רואים בגרף כמה חבילות TLS נשלחו לאורך זמן, כולל חיבורים חדשים (ClientHello, ServerHello).

צירי הגרף:
- **ציר ה-X**: מציין את הזמן לפי שניות.
- **ציר ה-Y**: מציין את כמות החבילות לאינטרוול (פרק זמן של שנייה אחת).

השוואה:
1. **Chrome** ו-**Firefox** (דפדפנים) – מציגים רמות גבוהות של תעבורת TLS בהתחלה, מה שנובע מפתיחת חיבורים מאובטחים חדשים לשרת. כלומר, הדפדפנים יוצרים הרבה Handshakes של TLS בתחילת ההקלטה, מה שמצביע על טעינת אתרי אינטרנט חדשים ב-HTTPS. כמו כן, בתהליך ה-Handshake מתבצעת גם החלפת מפתחות ההצפנה. בהמשך ההקלטה, לאחר ההתחברות הראשונית, הגרף יורד כי הדפדפן שומר על החיבורים פתוחים למניעת עומס ואין צורך ב-Handshake חדשים,
2. **Spotify** – ניתן לראות כי בתחילת החיבור מתבצע תהליך של חילופי מפתחות והסכמות הצפנה בקצב רב, אך הדבר לא נמשך זמן רב. ניתן להסיק שלאחר שקטע האודיו כבר נטען כפי שגם ראינו בגרפים הקודמים, התקשורת נשארת רציפה אך מתונה. כלומר לאחר שהקטע כבר נטען כמעט ולא מתבצעת העברת מפתחות בין הלקוח לשרת.
3. **YouTube** – אפליקציה זו מציגה פעילות נמוכה ביחס לדפדפנים מכיוון שיכול להיות שחלק מהנתונים שלו עוברים ב-UDP ולא ב-TLS. עם זאת, ל-YouTube, יש פרקי זמן מסוימים שבהם יש יותר חבילות, משום שהוא משתמש ב-Buffering - כאשר נטען קטע וידאו חדש, יש עלייה של TLS ואז ירידה.
4. **Zoom** – משתמש ב-TLS באופן מתמשך לאורך כל ההקלטה, כי הוא משתמש בהצפנה לכל אורך השיחה. כלומר, התעבורה כאן היא רציפה, לעומת לדוגמה YouTube, ששולח חבילות בפרק זמן קטן לפי צורך.

גרף ד' – Packet Size

הקדמה:
שלושת הגרפים המצורפים מציגים כיצד משתנה גודל החבילות עבור חמשת האפליקציות שלנו, לאורך זמן של כ-100 שניות. כל גרף מתייחס לטווח גדלים אחר:

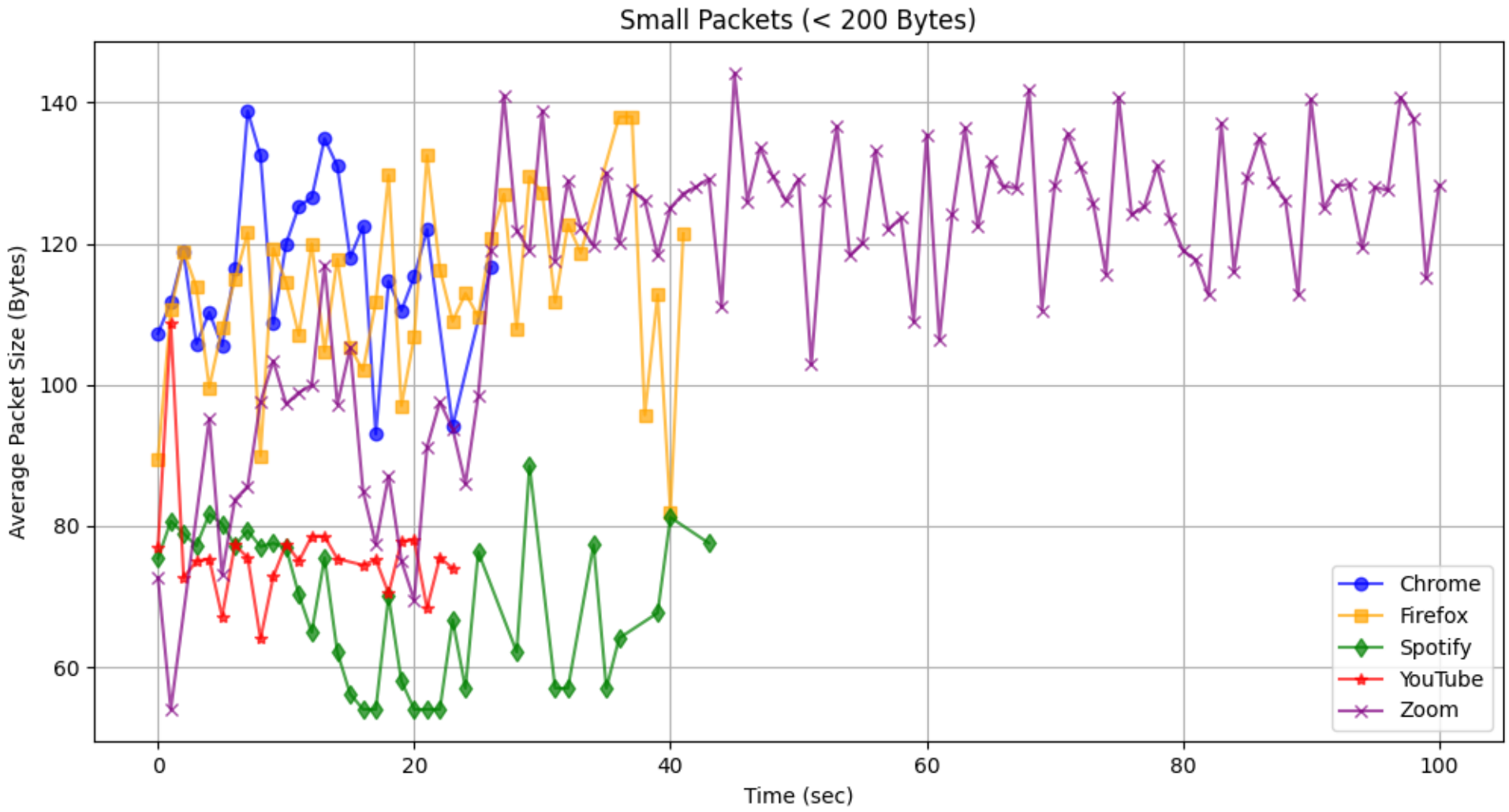
1. **חבילות קטנות** (חבילות שגודלן קטן מ-200 בתים)
2. **חבילות בינוניות** (חבילות שגודלן בין 200 בתים ל-1000 בתים)
3. **חבילות גדולות** (חבילות שגודלן מעל ל-1000 בתים)

באופן זה ניתן להשוות בצורה יותר נוחה בין האובייקטים בכל גרף.

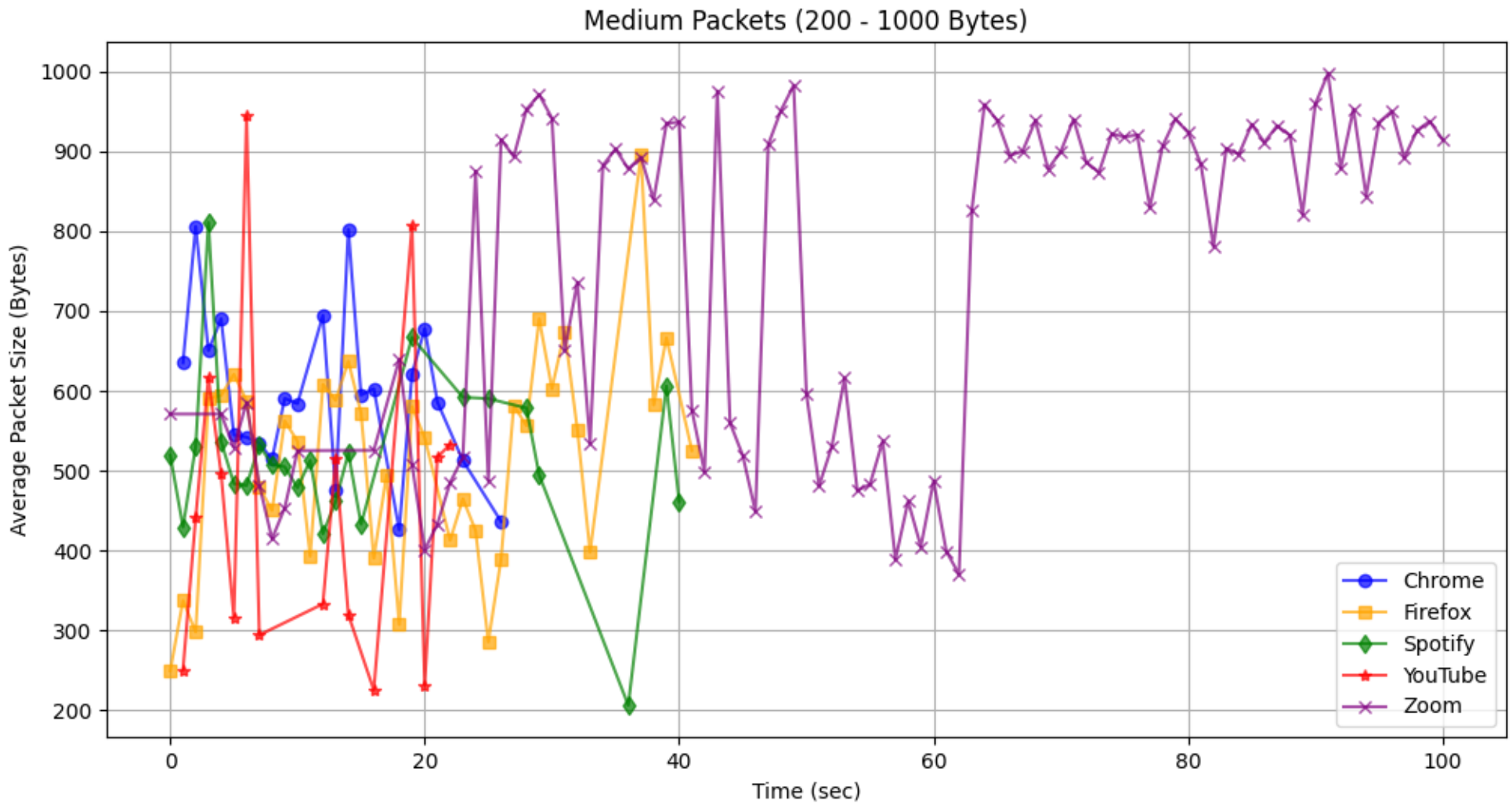
צירי הגרף:

1. **ציר ה-X**: מציין את הזמן לפי שניות.
2. **ציר ה-Y**: מציין את הגודל הממוצע של החבילה באותו רגע דגימה.

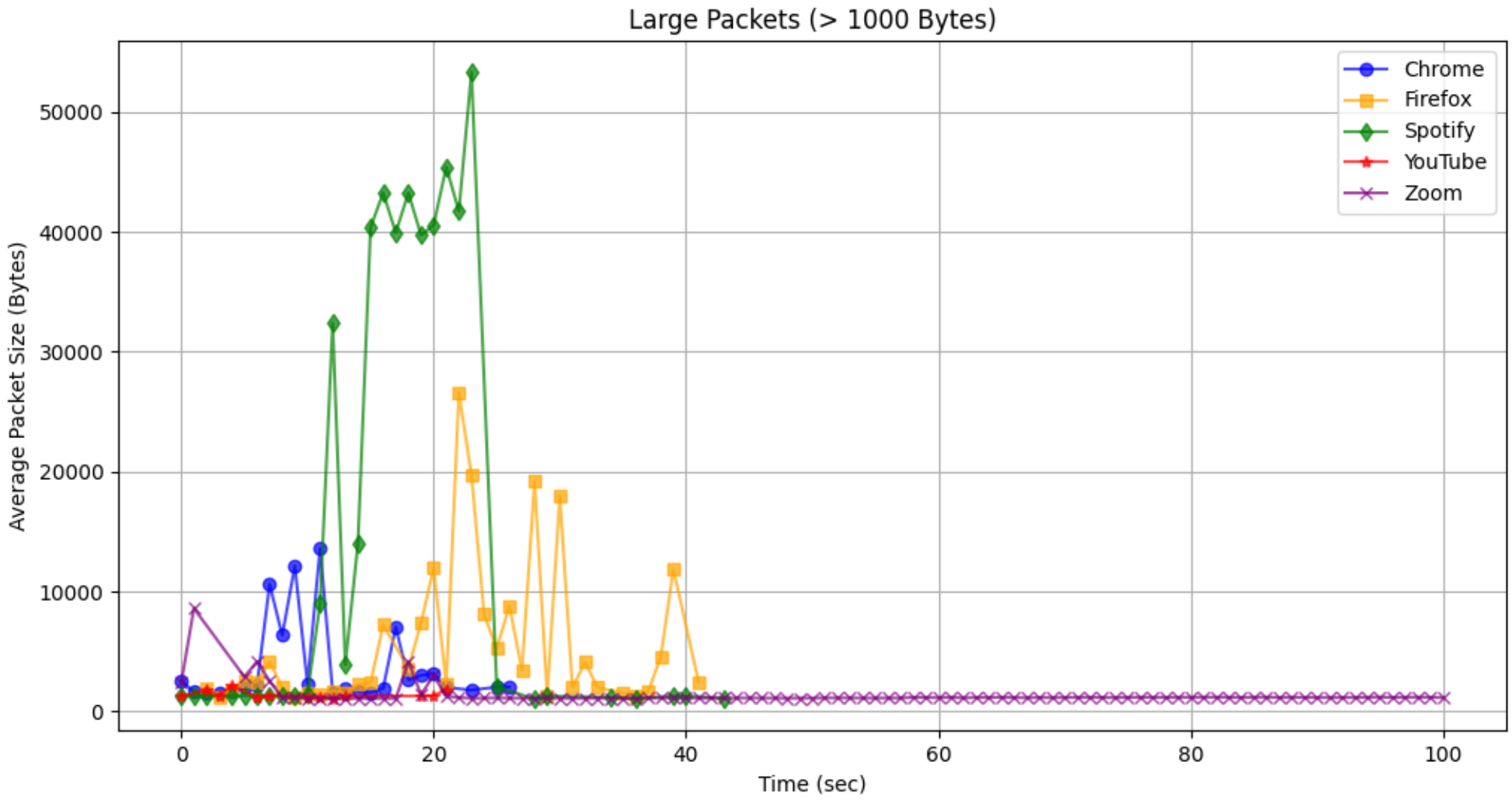
גרף ד' (1)
Small Packets (< 200 Bytes)



גרף ד' (2)
Medium Packets (200-1000 Bytes)



גרף ד' (3)
Large Packets (> 1000 Bytes)



גרף ד' – Packet Size

השוואה:

1. **Chrome** ו-**Firefox** (דפדפנים) – עבור חבילות קטנות ובינוניות אנו לא רואים איזה משהו שמאפיין את הדפדפים בצורה מיוחדת. לעומת זאת עבור החבילות הגדולות אני יכולים לראות שעבור שני הדפדפנים מתרחשות קפיצות חדות ופתאומיות. דבר הייחודי אך ורק עבור שני הדפדפנים.
מתוך כך אנו יכולים להסיק שהמידע שמועבר פה, מועבר אחת בכמה זמן בעזרת חבילה הגדולה משמעותית מהשאר, להשארתנו כאשר נטען עמוד אינטרנט חדש.
2. **Spotify** – עבור גרף חבילות הקטנות ניתן לראות שהחבילות קטנות יחסית, עם תנועות מתונות. בגרף החבילות הבינוניות ניתן לראות שלמעט מקטע בו כמות גודל החבילות היה נמוך ליחס לשאר השניות, כל שאר החבילות היו בטווח גודל יחסית קבוע ויציב. בחבילות הגדולות, ניצן לראות שהחבילות כאן תחילה בגודל משמעותית יותר גדול מאשר באפליקציות האחרות, אך לאחר מכן יש נפילה בגודל של גודל החבילות הגדולות.
מתוך כך ניתן להסיק שאפליקציה זו טוענת בבת אחת את כל קטע האודיו, ולאחר שהוא נטען במלואו גודל החבילות חוזר להיות מתון.
3. **YouTube** – ניתן לראות שעבור אפליקציה זו גודל החבילות הקטנות ברובו נמוך משמעותית מגודל החבילות של שאר האפליקציות. עבור החבילות הבינוניות ניתן לראות שגודל החבילות מכיל קפיצות חדות בקצב שמתרחש מספר פעמים.
מתוך כך ניתן להסיק שהזרמת וידאו מתרחשת ב"פיקים" כלומר אחת לכמה זמן מגיעה חבילה גדולה המכילה מידע גדול הקשור להזרמת הווידאו.
4. **Zoom** – עבור החבילות הקטנות אנו רואים שגודל חבילות אלו יותר גדול באופן יחסי לשאר החבילות גודלן תחילה אינו יציב ולאחר מכן גודל החבילות מתייצב. מצב דומה אנו יכולים לראות עבור גודל החבילות הבינוניות וגודל החבילות הגדולות. רק שבבינוניות גודלן מעל הממוצע ובגדולות מתחת.
מתוך כך ניתן להסיק שבמקרים של תקשורת רציפה הכוללת וידאו ואודיו אז תחילה לוקח לתקשורת להתייצב, אך לאחר זמן מה התקשורת מתייצבת ושומרת על גדלי חבילות בטווח יציב באופן יחסי לאפליקציות האחרות.

מסקנות מנקודת מבט של "תוקף" בעל גישה ל-Hash Flow ID

מתוך כך שהתוקף יודע את דפוסי התעבורה הכוללים גודל חבילות וחותמות זמן, הוא יוכל לדעת או לפחות להעריך באיזה סוג שירות "הנתקף" השתמש ובכך לשער מה הוא האתר עצמו בו "הנתקף" השתמש. הוא יוכל לדעת זאת לפי המידע שציינו במסקנות עבור הגרפים השונים.

- עבור גלישה בדפדפנים, במקרה שלנו Chrome ו-Firefox:
  כפי שראינו בגרפים השונים, ישנם עליות חדות במספר החבילות לפרק זמן מסוים, דבר שהסברנו שמתרחש כאשר נטען עמוד חדש או כאשר נטען תוכן חדש הכולל למשל תמונות.
  כפי שהסברנו וניתן לראות זאת גם בגרף המציג את גודל החבילות הגדולות (יותר מ-1000 Bytes), יש קפיצות פתאומיות בגודל החבילות. כפי שהסברנו, הדבר נובע מכך שברגעים אלו נטען דף חדש.

- עבור אפליקציות המשמשות להזרמת שירים (ללא וידאו), במקרה שלנו Spotify:
  ניתן לראות בגרף א' שמתקיים דפוס ראשוני של תעבורה יחסית מוגברת בקטע זמן מסוים ברגע בחירת הפודקאסט, ולאחר מכן כמות החבילות לפרק זמן מסוים יורדת משמעותית עד כדי כמעט ולא קורה.
  עבור גודל החבילות הקטנות והבינוניות, ניתן לראות שלמעט מקרה קיצון יחיד בגרף החבילות הבינוניות, רוב החבילות נשארות בגודל יחסית קבוע ויציב. עם זאת, עבור החבילות הגדולות ניתן לראות שיש עלייה משמעותית בגודל החבילות הגדולות (בעקבות הטעינה) ולאחר מכן נפילה בגודל החבילות הגדולות.

מסקנות מנקודת מבט של "תוקף" בעל גישה ל-Hash Flow ID

- עבור אפליקציות המשמשות להזרמת ווידאו, במקרה שלנו YouTube:
  ניתן לראות שמתקיים גם עבור כמות החבילות לפרק זמן וגם עבור גודל החבילות הקטנות והגדולות שתחילה קיימים "פיקים" ולאחר מכן הגרפים מתייצבים.
  עבור גרף החבילות הבינוניות, כפי שראינו מתרחשות קפיצות באופן תדיר. דבר שהסקנו שקורה בגלל שעבור הזרמת וידאו, מתרחש שאחת לכמה זמן מגיע מידע המכיל קטע מהסרטון (buffering).
- עבור אפליקציות לשיחות ווידאו, במקרה שלנו Zoom:
  לפי גרף א' ניתן לראות שמתקבלות מספר רב של חבילות לפרק זמן.
  עבור גרפים ב' ו-ג', ניתן לראות שתחילה יש עליה חדה במספר החבילות בתחילת ההתחברות, אך הדבר מתייצב לאחר שהחיבור נהיה יציב, ולאחר מכן כמות החבילות לפרק זמן נהיה יחסית קבוע.
  עבור כל גדלי החבילות מתקיים מצב דומה בו תחילה החבילות בגודל שאינו יציב ולאחר מכן גודל החבילות מתייצב.

מסקנות מנקודת מבט של "תוקף" בעל גישה ל-Hash Flow ID

ראשית, נשים לב שלתוקף קיימת גישה ל-Hash של ה Flow ID בלבד ולא לערכים עצמם המרכיבים את ה-tuple.
כלומר, לא קיימת לו גישה אל הערכים של רכיבי ה-Tuple הבאים:
source IP, destination IP, source port, destination port.
עם זאת, רק באמצעות ה- Hash, התוקף עדיין יוכל לבצע את הפעולות הבאות:

- **להבחין בין זרימות שונות.**
  הסיבה לכך היא שכל Hash מייצג חיבור נפרד.
- **להבין בין כמות החיבורים.**
  הסיבה לכך היא שבמידה ולתוקף יש את כתובת ה-Hash, אז התוקף יכול להבין כמה חיבורים המשתמש פתח, את אורך הזמן של כל חיבור והאם מדובר במספר חיבורים במקביל לאותו שרת או לשרתים שונים.
- **להשלים את התמונה הכוללת.**
  אם התוקף מכיר דפוסים של שימוש ספציפי (כפי שתואר בשלושת העמודים מעל), הגישה ל-Hash יכולה לסייע לו להשלים חלק מזיהוי השימוש של המשתמש.

מסקנות מנקודת מבט של "תוקף" בעל גישה ל-Hash Flow ID

לסיכום, על סמך דפוסי גודל החבילות, התזמון שלהן וכמות סוג הזרימות, התוקף יכול למפות איזה סוג אפליקציה פעלה (גלישה, שמע, וידאו, שיחת וידאו), כל זאת אף על פי שה-Flow ID מוסתר מאחורי Hash. בנוסף, התוקף יכול לנתח את דפוסי גודל החבילות והתזמון שלהן ובכך להסיק באיזה סוג שירות הלקוח משתמש ולהעריך באיזו אפליקציה הוא השתמש.

- עבור גלישה בדפדפנים:
  קיימים גלים לא סדירים בעת טעינה או מעבר לדפים חדשים.
- עבור אפליקציות המשמשות להזרמת שירים (ללא וידאו):
  תחילה קיימות עליות חדות, אך לאחר מכן נהיה קצב יציב, למעט בגודל החבילות הגדולות בהן יש עלייה לצורך טעינת קטע האודיו.
- עבור אפליקציות המשמשות להזרמת ווידאו:
  קיימות קפיצות בתחילת השימוש, לאחר מכן הקצב והגודל נהיה יחסית קבוע ונמוך, למעט בגרף גודל החבילות הבינוניות, בו קיימות קפיצות לצורך טעינת הסרטון.
- עבור אפליקציות לשיחות ווידאו:
  קיימות קפיצות חדות בגדלי החבילות בתחילת החיבור ולאחר מכן הגדלים מתייצבים. בנוסף מתקיים שלאורך כל שימוש השירות הזה, מתקבל ששירות זה שולח הכי הרבה נתונים באופן משמעותי.

מסקנות מנקודת מבט של "תוקף" <u>ללא</u> גישה ל-Hash Flow ID

כעת ננסה להבין מה מידע שהתוקף יכול לקבל כאשר אין לו גישה ל Hash Flow ID.
אנו ננסה להבין עד כמה אפשר לזהות באיזו אפליקציה המשתמש השתמש, כאשר המידע שנמצא בידי התוקף הוא גדלי החבילות וחותמות הזמן (timestamp) של החבילות השונות.

- נשים לב, גם אם התוקף לא יודע אילו מנות משתייכות לאיזה חיבור, הוא עדיין רואה מתי מגיעות חבילות ובאיזה גודל הן. לדוגמה, אם יש גלים חזקים של חבילות גדולות ואז הפסקה, זה עשוי להצביע על טעינת תוכן כבד. כלומר, כפי שכבר ציינו בעבר, הדבר יכול להעיד על טעינת דף אינטרנט חדש או טעינת סרטון וידאו.
דוגמה נוספת: אם התוקף רואה תחילה שתוכן החבילות אינו יציב ולאחר מכן מתייצב, זה עשוי להעיד על שימוש באפליקציה המשמשת לשיחת וידאו, כפי שהסברנו בשקופיות הקודמות.
- נשים לב שאף על פי זה שלתוקף אין גישה ל Hash Flow ID, התוקף עדיין מסוגל לראות זמנים של הגעת חבילות ולהסיק מכך מסקנות. בפרט, אם יש מספר חבילות המגיעות בזמן חופף או מחזור של חבילות בעלי דפוס דומה, הוא יוכל להעריך שמדובר בחבילות עם אותו Flow ID, מה שעוזר לו לדייק את מסקנותיו.

מסקנות מנקודת מבט של "תוקף" <u>ללא</u> גישה ל-Hash Flow ID

האם ניתן להבדיל בין אפליקציות שונות ללא גישה ל- Hash Flow ID?

- כפי שציינו במספר מקומות, ניתן לזהות או לכל הפחות לשער מהו סוג השירות בו המשתמש השתמש על סמך גדלי החבילות וחותמות הזמן בלבד. עם זאת, במידה והמשתמש משתמש במספר אפליקציות המספקות את אותו השירות, אין דרך לתוקף לדעת האם מדובר באפליקציה בודדת או באפליקציות מרובות.
  למשל, אם המשתמש משתמש באפליקציות YouTube ו-Netflix, התוקף יוכל לשים לב שמדובר בדפוסים המאפיינים אפליקציות שמספקות שירות של שידור וידאו, אך ככל הנראה יהיה לו קשה מאוד לדעת אם זו אותה אפליקציה או אפליקציות זרות, מפני שהפעם, בניגוד לתרחיש הקודם, הוא לא יודע אם ה-Hash Flow ID זהה או לא.
- בנוסף על כך, כאשר אין לתוקף גישה ל-Hash, יכול להיווצר אצלו בלבול. זאת משום שהתוקף לא יכול לשייך בין חבילה אחת לאחרת ובפרט, ייתכן שהדבר יטשטש את הדפוס של סיפוק תוכן כלשהו, או ששילוב החבילות ייראה כמו סיפוק תוכן אחר מאשר התוכן שהלקוח השתמש בו באמת.

לסיכום, אם המשתמש אכן השתמש רק באפליקציה בודדת, התוקף יוכל להעריך את סוג השירות בו הלקוח השתמש. עם זאת, כאשר המשתמש ישתמש באפליקציות מרובות, יהיה לתוקף קשה יותר להבין את הדפוס של החבילות וכך גם יהיה לו יותר קשה להבין בכמה אפליקציות המספקות את אותו שירות המשתמש השתמש.

בונוס
הסבר על ההקלטה

בהקלטה נכנסנו לדף האינטרנט של Spotify דרך דפדפן Chrome והפעלנו פודקאסט (ללא וידאו). במקביל, פתחנו את Gmail גם כן דרך Chrome ושלחנו מידי פעם מיילים. בסך הכל שלחנו שני מיילים – תוכן המייל הראשון היה קצר מאוד מבחינת כמות התווים שבו, לעומתו במייל השני כתבנו כמות גדולה יותר משמעותית של תווים.

חשוב לציין שבהקלטה זו, לעומת ההקלטות הקודמות, **ביטלנו את השימוש ב-QUIC** כדי לבודד משתנים שעשויים להשפיע על ניתוח התעבורה ובכך לדייק יותר את הניתוח שלנו. כמו כן, לפני שהתחלנו את ההקלטות ומכיוון שקיים שימוש ב-TLS, דאגנו לשמור קובץ מפתחות והשתמשנו בו לצורך פענוח התעבורה.

לאחר סיום ההקלטה, כדי לקבל הקלטה תמציתית ועם כמה שפחות "רעשים", ביצענו סינון של החבילות ב-Wireshark ע"י שימוש ב-Display Filter הבא:

```
(ip.addr == 35.186.224.0/24 || ip.addr == 142.250.0.0/16 || ip.addr == 64.233.160.0/19 || ip.addr == 66.249.80.0/20 || ip.addr == 216.58.192.0/19) && tcp.port == 443
```

הסינון הנ"ל ממקד את התעבורה שמקורה בטווח כתובות IP של שירותי Spotify ו-Gmail (אליהן הגענו באמצעות שימוש ב-nslookup spotify.com, nslookup mail.google.com ו-nslookup google.com, פקודה שלמדנו והשתמשנו בה גם במטלה 1), באמצעות פורט 443, שהוא פורט השימוש ב-HTTPS.
הסינון מבטיח שתעבור רק התעבורה שמגיעה משירותים אלו, תוך ביטול תעבורה שאינה רלוונטית ממקורות אחרים. בכך, קיבלנו שבהקלטה המסוננת, שעל בסיסה בנינו את הגרפים בשקופיות הבאות, בסה"כ יש 1859 חבילות.

הסבר על הסקריפט

- **plots_bonus.py**

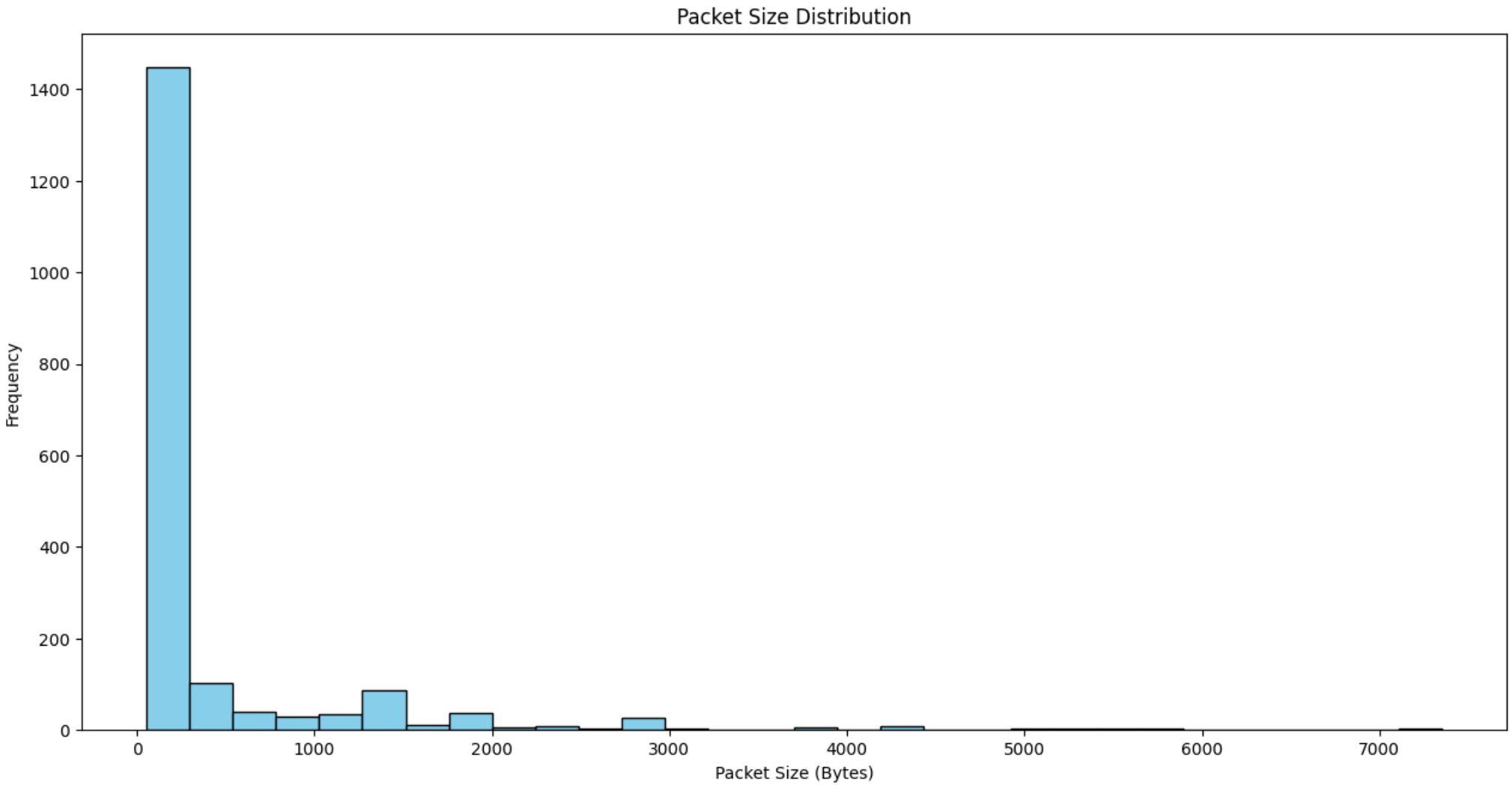
הסקריפט מנתח את ההקלטה שביצענו (קובץ pcap), בכך שהוא מחלץ מידע על תעבורת הרשת ומציג אותו בצורה גרפית. הסקריפט מבצע חמישה סוגי ניתוחים (פירוט בהמשך המצגת):

1. התפלגות גודל החבילות
2. הפרשי זמן בין חבילות
3. נפח זרימה (Bytes לכל זרימה)
4. גודל הזרימה (מספר חבילות לכל זרימה)
5. כתובות IP יעד נפוצות

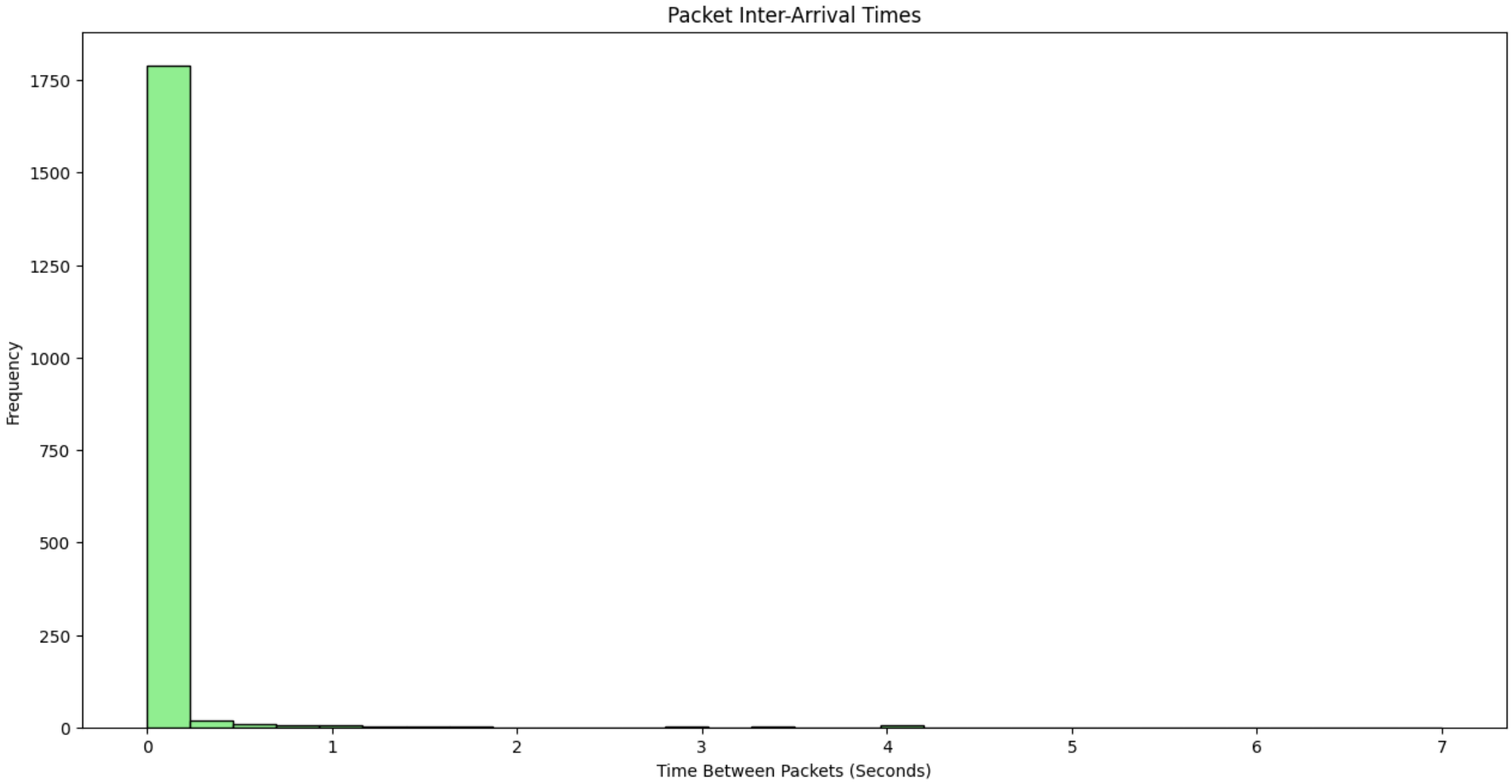
הסקריפט משתמש בספריית pyshark לקריאת קובץ ההקלטה וב-matplotlib ליצירת הגרפים.

כדי להריץ את הסקריפט, יש לוודא כי מזינים את הניתוב ושם הקובץ של ההקלטה במשתנה pcap_file המוגדר בראש הסקריפט.

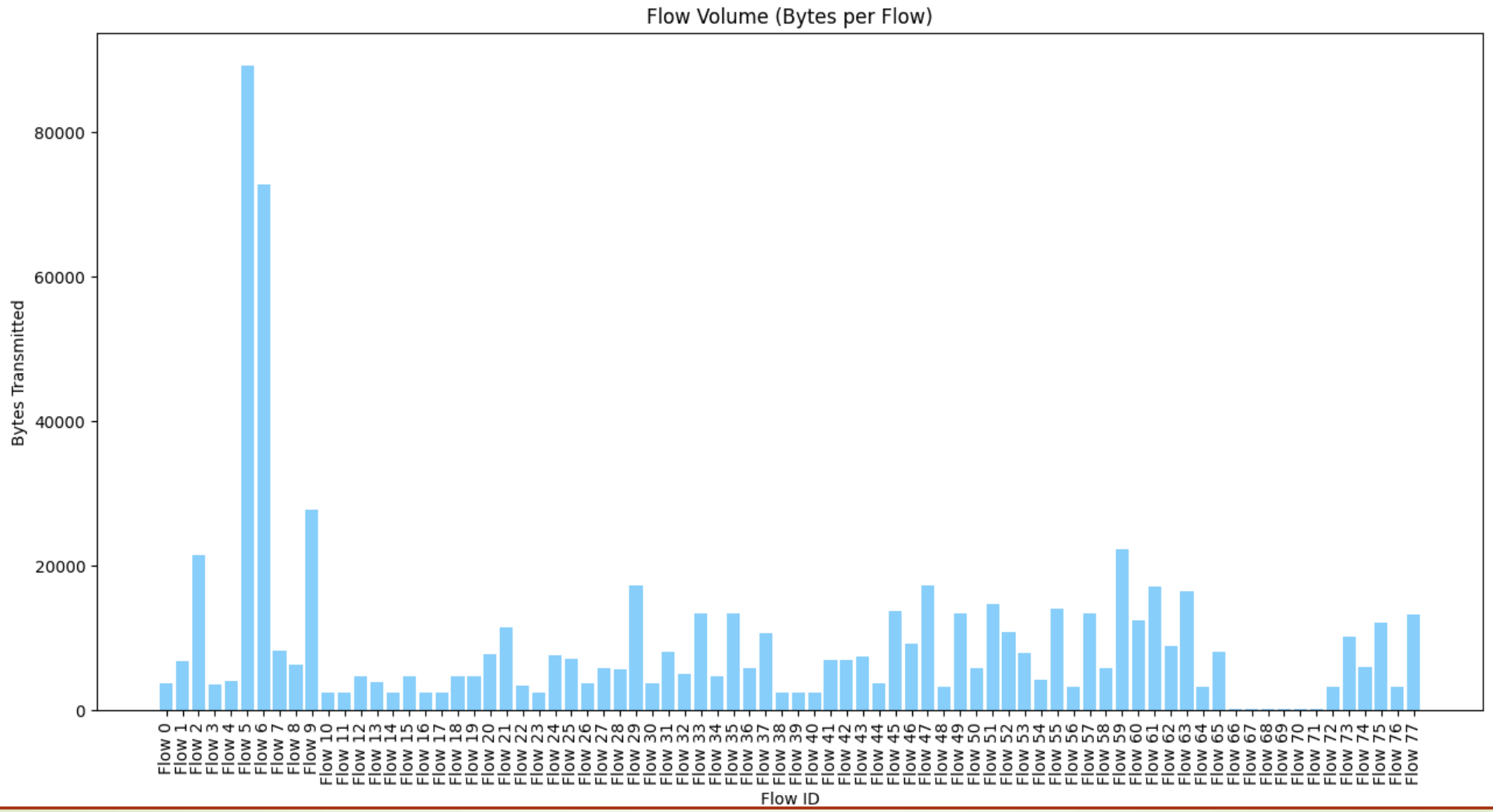
גרף 1 – התפלגות גודל החבילות



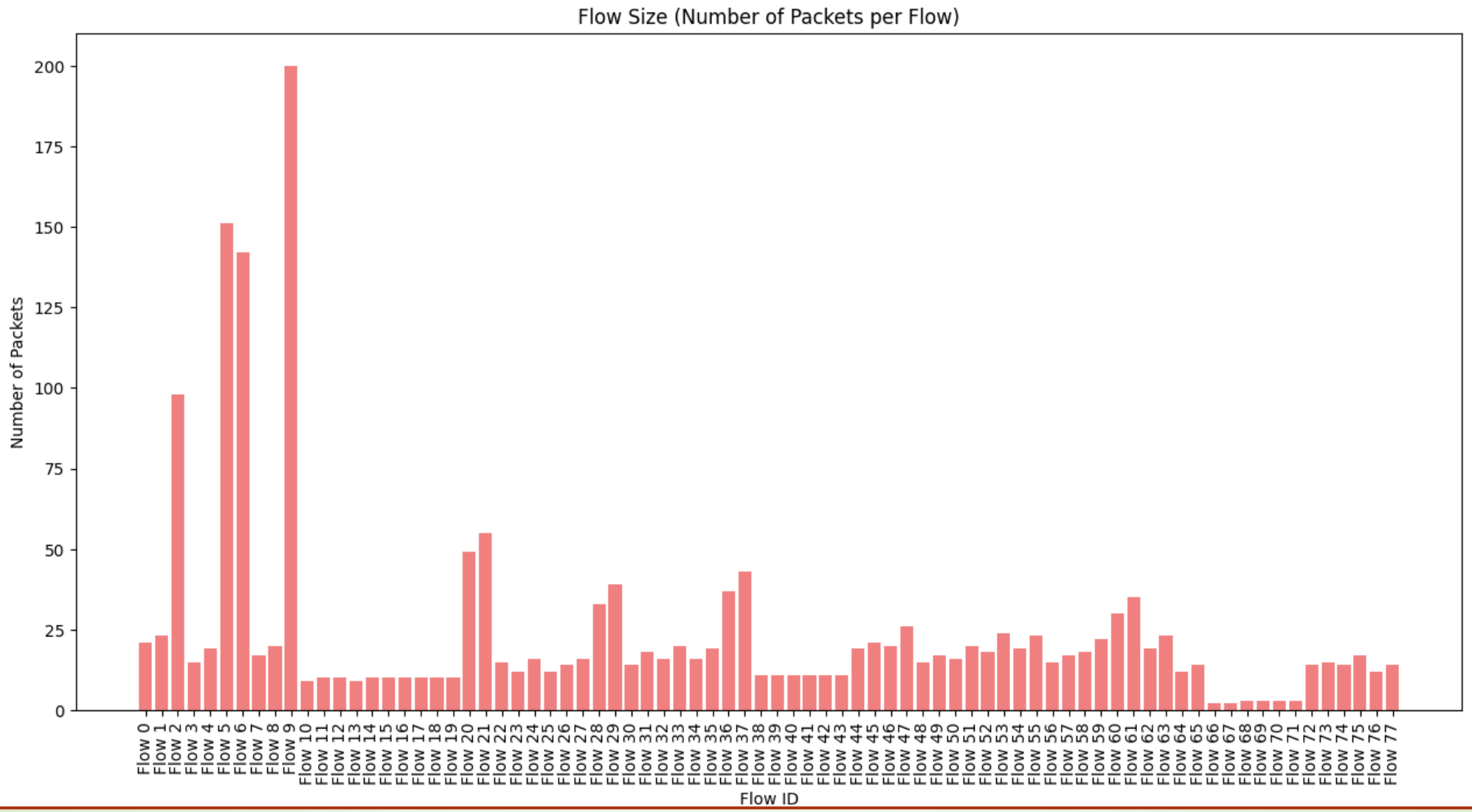
גרף 2 – הפרשי זמן בין חבילות



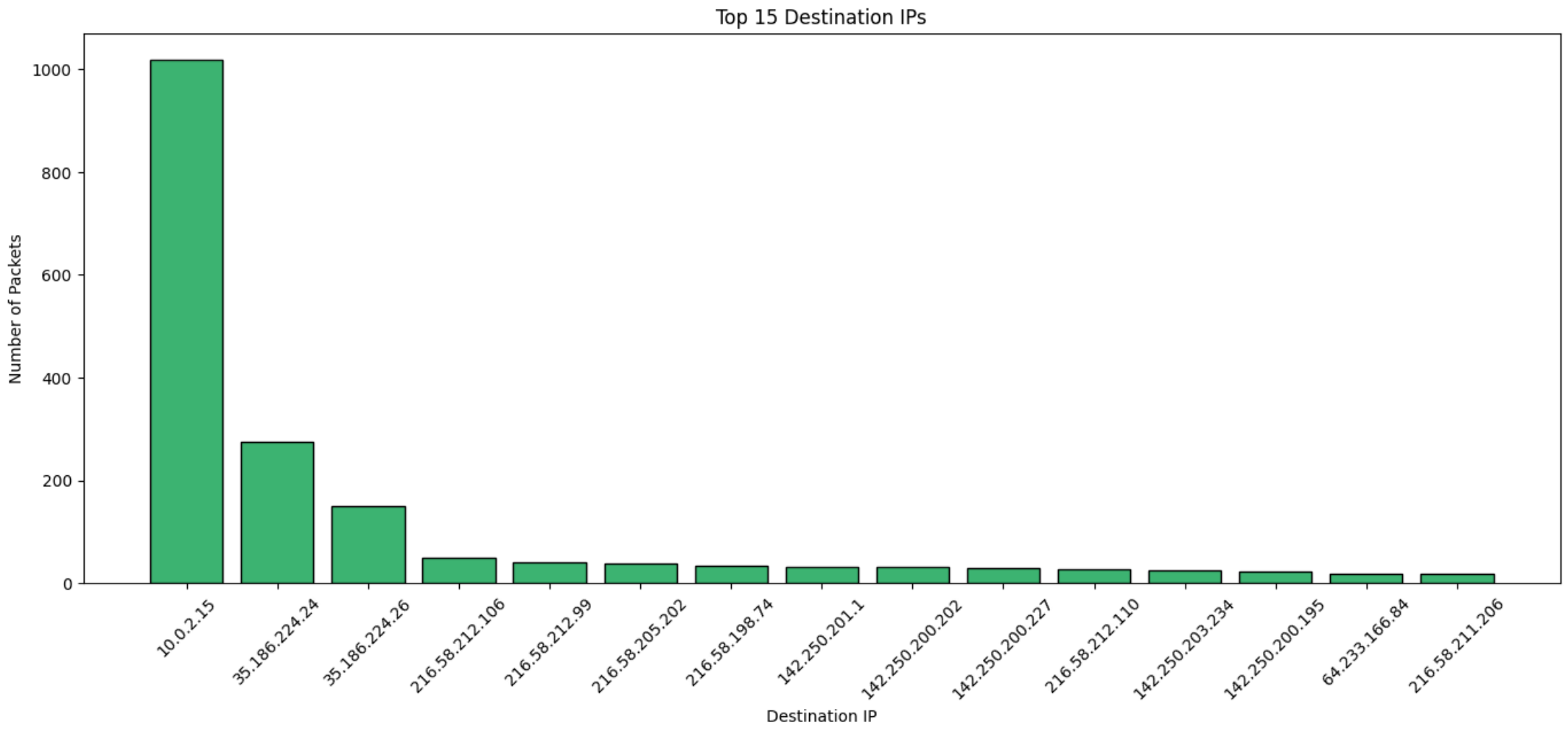
גרף 3 – נפח זרימה (Bytes לכל זרימה)



גרף 4 – גודל הזרימה (מספר חבילות לכל זרימה)



גרף 5 – כתובות IP יעד נפוצות



ניתוח התעבורה ומסקנות העולות מהגרפים

גרף 1 – התפלגות גודל החבילות:
הגרף מראה שתעבורת הרשת במהלך ההקלטה הייתה מורכבת בעיקר מחבילות קטנות. זה אכן מתאים לפעולות שביצענו – האזנה לאודיו ושליחת מיילים, שתי הפעולות לא יוצרות כמות גדולה של תעבורה רציפה, לעומת לדוגמה סטרימנג של ווידאו. הגרף לא מאפשר להבדיל בדיוק בין Spotify ל-Gmail, מכיוון ששניהם יכולים לייצר חבילות קטנות. עם זאת, חבילות גדולות יותר (מעל ל-1000 בתים) ככל הנראה קשורות לחלקים מסוימים של תעבורת הדפדפן שגלשנו דרכו (Chrome) או הזרמת המוזיקה עצמה.

גרף 2 – הפרשי זמן בין חבילות:
הגרף מציג את הפרשי הזמן שעוברים בין הגעת חבילות. הוא מראה כמה פעמים חלף זמן מסוים בין קבלת שתי חבילות רצופות. רוב החבילות מגיעות תוך פרק זמן קצר מאוד זו אחרי זו, זה בא לידי ביטוי בגרף בכך שיש שיא חד בהתחלה. כמו כן, יש מספר קטן של חבילות שמגיעות עם הפרש זמן גדול יותר. ההתנהגות הזו אכן מתאימה לעובדה ש-Spotify שולחת הרבה חבילות בקצב יציב (סטרימנג של מוזיקה). ההפסקות הקטנות יכולות להיות תוצאה של תעבורה כמו טעינת דף אינטרנט או Gmail ברקע. בנוסף, באשר לערכים הגבוהים יותר בגרף, שמופיעים לעיתים נדירות, ייתכן שמתקשרים לכך ששלחנו מידי פעם מיילים.

גרף 3 – נפח זרימה (Bytes לכל זרימה):
ראשית, זרימה מתארת סדרה של חבילות השייכות לאותה תקשורת בין שני צדדים. הגרף מראה את נפח הזרימה, כלומר את מספר הבתים שהועברו בכל זרימה. כל עמודה מייצגת זרימה אחרת, והגובה שלה מראה כמה בתים הועברו בזרימה הזו. הזרימות הגדולות כנראה שייכות ל-Spotify – זרם יציב של נתונים לצורך השמעת מוזיקה. כמו כן, Gmail ככל הנראה בא לידי ביטוי בזרימות הקטנות יותר, שכן מיילים לרוב צורכים פחות תעבורה.

ניתוח התעבורה ומסקנות העולות מהגרפים

גרף 4 - גודל הזרימה (מספר חבילות לכל זרימה):
הגרף מציג את גודל הזרימה, כלומר את מספר החבילות בכל זרימה. כל עמודה מייצגת זרימה אחרת (לכל זרימה יש מזהה – Flow ID), והגובה שלה מראה כמה חבילות היו בזרימה הזו. הדפוס בגרף דומה לגרף הקודם (גרף 3) – יש זרימות בודדות עם הרבה חבילות ורובן עם מעט חבילות. לפיכך, הגרף מחזק את מה שראינו בגרף הקודם, שכן מספר מצומצם של זרימות מייצר את רוב התעבורה. כמו כן, הזרימות עם הרבה חבילות הן כנראה אותן זרימות שנראו גדולות מבחינת הנפח (Spotify).

גרף 5 - 15 כתובות IP יעד נפוצות:
הגרף מראה את 15 כתובות ה-IP של היעד, שאליהן נשלחו הכי הרבה חבילות במהלך ההקלטה. הוא מציג עם אילו שרתים הייתה הכי הרבה תקשורת. קל לראות שהכתובת 10.0.2.15 מובילה בפער עצום, זו כתובת פנימית (localhost) של המחשב שממנו ביצענו את ההקלטה. כמו כן, אחריה מופיעות כתובות ציבוריות כמו:
35.186.224.26, 35.186.224.24 – כתובות IP הקשורות ל-Spotify.
216.58.x.x – כתובות IP של שרתי Google.
142.250.x.x – כתובות IP של שרתי Google, בפרט של Gmail.
64.233.166.85 – ככל הנראה כתובת IP נוספת שקשורה ל-Google.

מקורות שהשתמשנו בהם

1. MPTCP:
https://chatgpt.com/share/67e43aa1-4260-8012-b2ee-3cb3b461d72a

2. TLS Keys:
https://www.youtube.com/watch?v=5qecyZHL-GU&ab_channel=ChrisGreer

3. כיבוי QUIC בדפדפן Chrome:
https://chatgpt.com/share/67e31ec5-5468-8007-a891-d3f4555a0d7e

4. Traffic Flow:
https://en.wikipedia.org/wiki/Traffic_flow_(computer_networking)

5. זיהוי כתובות IP (בונוס):
https://chatgpt.com/share/67e444ec-0660-8007-aa56-cdd790e88a26