

# רשתות תקשורת - פרויקט גמר

**קבצי ההקלטות נמצאים כאן:**

[https://drive.google.com/drive/folders/1mz\\_QXR5QImiu4UvDIwEPCSz7vvq817\\_4?usp=drive\\_link](https://drive.google.com/drive/folders/1mz_QXR5QImiu4UvDIwEPCSz7vvq817_4?usp=drive_link)

תאריך ההגשה: 05.03.2025

## תוכן עניינים:

חלק:	עמודים:
חלק 1	<a href="#">3-9</a>
חלק 2	<a href="#">10-22</a>
חלק 3:	<a href="#">23-45</a>
בנוס:	<a href="#">46</a>
נספח 1:	47-48

## חלק 1: מענה על שאלות

1. **משתמש מדווח שהעברת הקבצים שלו איטית, ועליך לנתח את שכבת התעבורה כדי לזהות את הסיבות האפשריות. אילו גורמים עשויים לגרום להעברה האיטית, וכיצד היית פותר את הבעיה?**

**1.1. בעיה:** שימוש בפרוטוקול UDP להעברת חבילות עלול לגרום לאובדן נתונים, שכן UDP אינו כולל מנגנון שידור חוזר כמו TCP ואינו מספק בקרת זרימה לשליטה על קצב השידור. כתוצאה מכך, אם חבילות אובדות או מגיעות בסדר שגוי, אין תיקון אוטומטי, מה שעלול להוביל להאטה או לתקלות בהעברת הנתונים.

**פתרון:** יש לבדוק האם ניתן להמיר את ההעברה ל-TCP, שמבטיח אמינות באמצעות מנגנוני אישור (ACKs) ושידור חוזר במקרה של אובדן חבילות. TCP גם כולל בקרת זרימה שמונעת הצפת הרשת ושומרת על יציבות הביצועים.

לחילופין, אם השימוש ב-UDP נדרש לצורך מהירות והשהיה נמוכה, ניתן לעבור ל-QUIC, שמבוסס על UDP אך מכיל תיקון שגיאות ובקרת זרימה, ובכך משלב ביצועים גבוהים עם אמינות טובה יותר.

**1.2. בעיה:** כאשר TCP מזהה שאיבוד חבילות (Packet Loss), הוא משדר אותן מחדש, מה שעלול להאט את קצב ההעברה. ככל שאובדן החבילות גבוה יותר, כך יהיו יותר שידורים חוזרים, מה שיוביל לעיכובים נוספים ויפגע ביעילות ההעברה.

**פתרון:** אחת הדרכים לצמצם איבוד חבילות היא להשתמש בחיבור קווי (Ethernet) במקום Wi-Fi, שכן חיבור קווי יציב יותר, מספק זמן השהיה (Latency) נמוך יותר, מפחית משמעותית את הסיכוי לאובדן חבילות ואינו מושפע מהפרעות חיצוניות כמו אותות אלחוטיים מתנגשים בנוסף, ניתן להגדיל את גודל חלון ה-TCP (TCP Window Scaling) כדי לשפר את קצב ההעברה. מנגנון זה מאפשר לשלוח לשלוח יותר נתונים לפני המתנה לאישור (ACK), ובכך מפחית את כמות ההפסקות בשידור ומייעל את ניצול רוחב הפס.

**1.3. בעיה:** שליחת הודעות מרובה: שליחת הודעות מרובות ללא שליטה, מה שיכול להוביל לאיבוד חבילות ובזבוז זמן עיבוד.

**פתרון:** נגביל את קצב השליחה בפרוטוקול ה-TCP או באפליקציה עצמה, מה שמונע עומס ומגביל את קצב העברה, כל פעם שנגיע למכסת ההודעות שניתן לשלוח (לדוגמא בשנייה), פרוטוקול ה-TCP ימתין עד לקבלת כל ה-ACKs ולאחר מכן ניתן להמשיך.

**1.4. בעיה:** גודל חלון ב-TCP : TCP ישנו שימוש ב Sliding Window בכדי שבעת איבוד נתונים לא ישלחו יותר מדי נתונים מבלי לקבל ACK. אך , אם נגדיר את גודל החלון להיות קטן מדי נצטרך לחכות לעיתים תכופות מדי לאישור מה שיצור העברה איטית גם אם אכן יש רוחב פס גבוה.

**פתרון:** קודם כל יש לזהות את מקור הבעיה על ידי ניתוח החבילות ב Wireshark, ל לאחר שהבנו את מקור הבעיה נוכל להגדיל את חלון ההעברה מהצד של השולח ואם אפשרי גם המקבל בכדי שנוכל לשלוח יותר נתונים טרם קבלת האישור.

**1.5. בעיה:** עיכוב בהגעת הקבצים עקב זמן תגובה גבוה מצד השרת (RTT)-  
במידה וה-RTT יהיה ארוך , המשתמש יצטרך לחכות הרבה זמן בכל פעם שיקבל ACK מהשרת , וזה יגרום לקצב השידור להיות איטי יותר.

**פתרון:** נבדוק את נתיב השרת בעזרת traceroute או ping כדי לנתח את זמן התגובה של שרת ואופן ניתוב המידע ברשת . אם נקבל זמן תגובה גבוה, נוכל לנתב מחדש את נתיב השרת כך שהמידע תעבור דרך שרת חלופי ומהיר יותר. זו יכולה להיות גם בעיה גיאוגרפית, לכן יהיה עלינו לעבור לשרת קרוב יותר. בנוסף נוכל לשלב שימוש בפרוקסי עם cache (למניעת בקשות חוזרות).

2. נתח את ההשפעות של מנגנון בקרת הזרימה של TCP על העברת נתונים. כיצד הדבר ישפיע על

הביצועים כאשר למקור יש כוח עיבוד גבוה משמעותית מזה של המקבל?

2.1 בקרת הזרימה (Flow Control) בפרוטוקול TCP מיועדת למנוע מהשולח להציף את המקבל בכמות נתונים שהוא אינו מסוגל לעבד בזמן אמת, ע"י שימוש במנגנונים הבאים: Sliding window, Receiving window, ACK send/receive.

**הבעיה:** כאשר המקור בעל יכולת עיבוד גבוהה מיכולת העיבוד של היעד, יוצר חוסר איזון המוביל למספר דברים:

כאשר המקור בעל כוח עיבוד גבוה מכוח העיבוד של היעד, הוא ישלח הודעות בקצב גבוה, מה שימלא את קיבולת הbuffer בצד המקבל.

כאשר הbuffer מתמלא, גודל הreceiving window יקטן עד כדי "0", מה שמוביל לכך שהשולח ימתין עד לפינוי הbuffer, ויגרום להאטה בקצב העברת הנתונים.

הצד המקבל, שלא מצליח לעבד במהירות את הנתונים המתקבלים, מתקשה לשלוח ACKs בתדירות גבוהה.

במקרים מסוימים, אם העיכוב נמשך, ה-Sliding Window אף עלול להצטמצם, מה שמגביל עוד יותר את קצב ההעברה.

אם השולח אינו מקבל ACK בזמן, הוא עלול לפרש זאת כהודעה שאבדה או כעומס ברשת ולבצע שליחה חוזרת (Retransmission), וכתוצאה מכך, כמות הנתונים המתקבלת אצל המקבל עולה עוד יותר, מה שמכביד על הבאפר ומחמיר את הבעיה.

3. בחירת המסלול משפיעה על ביצועי הרשת בכמה אופנים , נסביר את העיקריים שבהם ונגדיר

גורמים אותם יש לקחת בחשבון בעת קבלת החלטות ניתוב:

**3.1. עומס ברשת - congestion:**

אם ישנם מספר גדול של חיבורים שמשתמשים באותו מסלול , יכולים להיגרם עומסים, בעקבות עומסים אלו יכולים להיווצר איבודים של חבילות ובעקבותיהם גם שידורים חוזרים שיאטו עוד יותר את החיבור בעקבות בחירת מסלול זה.

**3.2. אורך המסלול , כמות הנתבים בדרך וזמן השהיה - latency:**

נגדיר זמן השהיה - הזמן שעובר מהרגע שבו חבילה נשלחת ממקור ועד לרגע שבו היא מגיעה ליעדה. כאשר נשתמש במסלול ארוך יותר עם הרבה hops (נתבים) בדרך החבילות יגיעו באיחור מה שיפגע בביצועים של הרשת. אך , כאשר נשתמש במסלול עם כמה שפחות נתבים ונוכל לבצע קישורים מהירים זמן השהיה יהיה נמוך יותר , לכן נשתדל לבחור בשרתים יותר קרובים ופחות עמוסים כך שזמן השהיה יהיה נמוך יותר. (לא תמיד , נראה עוד שיקולים בהמשך)

**3.3. רוחב פס זמין - bandwidth:**

רוחב הפס גם הוא משפיע על זמן ההעברה, כל מסלול עשוי לכלול קישורים עם רוחב פס שונה , לדוגמא, אם מסלול מסוים עובר דרך נתבים בעלי חיבור איטי או מהירות איטית מה שעלול לקרות בדרך הוא מעין "צוואר בקבוק" שיגרום להאטה בתהליך העברת החבילות.

**3.4. אמינות - reliability:**

לכל מסלול יש עמידות שונה לכשל. במידה נתב או קישור נופלים, המסלול צריך להיות מוחלף במסלול גיבוי במהירות האפשרית. פרוטוקולים כמו OSPF ו-BGP יכולים להבטיח החלפת מסלול אוטומטית במקרה של כשל.

#### 4. כיצד פרוטוקול MPTCP משפר את ביצועי הרשת?

Multipath TCP מאפשר העברת נתונים דרך מספר מסלולים במקביל, במקום העברתם במסלול יחיד. ע"י חלוקת הנתונים דרך sub-flows (תת חיבורים), מנוצל רוחב הפס הזמין באופן מיטבי.

##### 4.1 הגדלת קצב העברה

בעת שימוש ב-TCP, כל החבילות עוברות דרך נתיב אחד בלבד, כך שמהירות ההעברה מוגבלת לרוחב הפס של אותו נתיב. MPTCP מאפשר שימוש במספר חיבורים במקביל כך שהתעבורה מתחלקת בין כמה ערוצים באותו הזמן, ובכך גם מעלה את קצב ההעברה הכולל. לדוגמא: ע"י חיבור ב Wifi ורשת סלולרית (4G או 5G) במקביל, או חיבורים דרך מספר נתבים או שרתים במקביל. כתוצאה מכך ישנו יותר רוחב פס זמין, הנתונים יכולים לעבור דרך מספר מסלולים בו זמנית ובכך הנתונים שיעברו ביחידת זמן מסוימת יהיו גדולים יותר מ-TCP.

##### 4.2 שיפור אמינות וחוסן הרשת

בניגוד ל-TCP שבו קיים רק מסלול אחד להעברת הנתונים, שבמידה ומשהו נכשל באותו המסלול נצטרך להפסיק את ההעברה עד למציאת מסלול חדש, ב MPTCP ההעברה מבוצעת ע"י מספר מסלולים באותו הזמן, כך שגם אם יש בעיה באחת הנתיבים או שהעברה בהם נכשלת, ההעברה תמשיך להתבצע בשאר הנתיבים ללא עיכוב או הפסקה. בכך בעת כשל ברשת לא תהיה השפעה משמעותית כמו ב-TCP על ביצועי הרשת והמידע יוכל להמשיך לעבור.

##### 4.3 זמן שהיה קטן יותר

בעת שימוש ב-MPTCP תתבצע בחירה של המסלול האופטימלי באופן דינמי, כאשר ישנה התחשבות בזמן שהיה, בעומסים, וביכולת של ה hop הנוכחי והחיבור של המשתמש. אם אחד הנתיבים עמוס או איטי, התעבורה מנותבת לנתיבים מהירים יותר בזמן אמת.

5. אתה מנטר תעבורת רשת ושם לב לאובדן חבילות גבוה בין שני נתבים. נתח את הסיבות האפשריות לאובדן חבילות בשכבות הרשת והתעבורה, והצע צעדים לפתרון הבעיה.

#### 5.1. סיבות לאובדן חבילות בשכבת הרשת:

##### 5.1.1. עומס ברשת:

כאשר אחד הנתבים נמצא בעומס תעבורה, הוא מתחיל להיפטר מחבילות קיימות עקב מחסור בזכרון של buffer, בעיה זו מתרחשת כאשר קיבלת הרשת נמוכה או ברשתות שלא מקיימות ניתוב אופטימלי.

**פתרון:** נזהה את עומסי הרשת ומהיכן הם נגרמים, ניתן עדיפות לתעבורה הקריטית ונשתמש בפרוטוקולים המשפרים עומסים כמו: load balancing, הבודק האם יש מספר נתיבים אפשריים מאותו מקור לאותו יעד אז יבדוק איפה ניתן לפזר את העומס כדי למנוע ריכוז חבילות במסלול יחיד\ לא אופטימלי. בנוסף אם העומס גבוה ניתן לשדרג את הנתב או את רוחב הפס.

##### 5.1.2. בעיות ניתוב (Routing Issues):

יש כמה גורמים לבעית הניתוב:

##### 5.1.2.1. כתובת יעד שגוי

**פתרון:** לוודא שהכתובת תקינה לפני שאנחנו מתקשרים איתו או לעדכן דרך נתיב אופטימלי או באמצעות ping או traceroute.

##### 5.1.2.2. הבעיה שנכנסים ללולאות ניתוב (Routing Loop) וזה קורה שהחבילה

עוברת בין שני נתבים ללא כתובת יעד.

**פתרון:** להשתמש ב מנגנונים למניעת הבעיה הזאת כמו OSPF

##### 5.1.2.3. נתיב לא עדכני (Stale Route) כאשר יש קשר עדיין קיים בטבלה אבל

הקשר כבר נותק זה גורם לשליחת חבילות לנתיב שאינו פעיל

**פתרון:** להשתמש בפרוטוקול ניתוב דינמי כמו: OSPF, BGP שיעדכן את הטבלה באופן אוטומטי או בהגדרת timeout עבור נתיבים ישנים הקיימים בטבלה.

##### 5.1.3. TTL נמוך (Time to Live Expired) – אם ערך ה-TTL של חבילה נמוך מדי,

היא תימחק לפני שתגיע ליעד.

**פתרון:** הגדלת ערך TTL של החבילות אם הבעיה מתרחשת בתחנות קצה (כמו מחשבים או שרתים), יש לוודא שה-TTL של החבילות היוצאות גבוה מספיק כדי להגיע ליעדן. בדוק כמה קפיצות (Hops) יש בין המקור ליעד. אם הרשת כוללת מספר רב של נתבים, יש להגדיל את ה-TTL בהתאם.

##### 5.1.4. שרת DNS לא מעודכן (בעיות בטבלת הניתוב) – אם טבלאות הניתוב אינן

מעודכנות, הנתבים עלולים לנתב חבילות לכתובות שגויות או להשתמש במסלולים עמוסים ולא אופטימליים.

**פתרון:** עדכון טבלת ה DNS ע"י בדיקת הניתוב כדי לראות אם הכתובת אכן נגישה, ניקוי המטמון לאחר תקופה מסויימת של זמן או שימוש בשרת DNS אמין כמו google, cloudflare.



## 5.2.

### סיבות לאובדן חבילות בשכבת התעבורה:

#### 5.2.1. נתק חיבור TCP - אם חיבור ה-TCP מתנתק עקב Timeout או עומס יתר, הנתב

השולח עשוי שלא לקבל אישור קבלה בזמן ולהפעיל מנגנון ניתוק. במקרה כזה, החיבור כולו ייסגר, וכל הנתונים שלא הועברו בהצלחה יאבדו, מה שעלול לגרום להאטה וביצועים ירודים ברשת.

**פתרון:** יש להגדיל את זמן ה-timeout בפרמטרים של tcp כדי לאפשר התאוששות טובה יותר מעומס זמני ולהשתמש במנגנונים כמו keep-alive ו-fast retransmit כדי למנוע ניתוקים לא רצויים. בנוסף, ניתן ליישם Qos כדי לתת עדיפות לחיבורים חשובים ולהפחית עומס על הרשת.

#### 5.2.2. שימוש בפרוטוקול UDP - UDP אינו מספק מנגנון אישור קבלה, ולכן חבילות

עלולות ללכת לאיבוד מבלי שהשולח ידע זאת.

**פתרון:** שימוש בפרוטוקול QUIC אשר מעל UDP אשר מספק מנגנון ACK כמו ל-TCP.

#### 5.2.3. פרגמנטציה ושחזור חבילות - אם חבילת TCP גדולה מגודל ה-MSS של המקבל,

היא עשויה להידחות, מה שיגרום לאובדן נתונים. בנוסף, אם חבילה אחת מתוך מקטעים נאבדת, TCP ישלח מחדש את כל הזרם, מה שמוביל לעומס והשהיות ברשת.

**פתרון:** יש להגדיר MSS מתאים בהתאם ל-MTU של הרשת ולהפעיל Path MTU Discovery למניעת פרגמנטציה. בנוסף, שימוש ב-Selective Acknowledgment יאפשר ל-TCP לשלוח מחדש רק את החבילות שאבדו במקום את כל הזרם.

#### 5.2.4. חלון קבלה קטן מדי - אם הנתב המקבל אינו מעבד נתונים מספיק מהר, הוא עשוי

לשלוח עדכון לחלון הקבלה עם גודל קטן מאוד או אפילו אפס. מצב זה יכול לגרום לנתב השולח לעצור זמנית את שליחת הנתונים, ובמקרים של עיכובים או שגיאות ברשת, חבילות עלולות להישמט או להתיימן לפני שהן נשלחות מחדש.

**פתרון:** הגדלת גודל חלון הקבלה באופן דינמי ושימוש במנגנון window scaling מאפשרים ניצול טוב יותר של רוחב הפס ומונעים השהיות ואובדן חבילות.

## חלק 2: קריאת שלושה מאמרים ומענה על השאלות:

יש לענות על השאלות הבאות עבור כל מאמר :

**מהי התרומה המרכזית של המאמר?**

- איזה חידוש או ערך עיקרי הוא מציג?

**באילו מאפייני תעבורה המאמר משתמש, ואילו מהם הם חדשים?**

- האם יש תכונות שלא הופיעו במאמרים קודמים?

**מהן התוצאות המרכזיות (ניתן להעתיק את התרשימים מהמאמר), ומה המסקנות העיקריות מהתוצאות הללו?**

- אילו תובנות עולות מהתוצאות, ואיך הן משפרות את ההבנה בתחום?

## מאמר מספר 1 :

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

### מהי התרומה המרכזית של המאמר?

המאמר בראשיתו דן בסוגיית החיפוש אחר פתרון של ניתוח נכון ומדויק של תעבורה מוצפנת ברשת. הוא מציג מספר שיטות המשמשות כיום, ומסביר מדוע כל אחת מהן אינה רלוונטית עוד עקב השתכללות ההצפנה ההולכת ומשתפרת של תעבורת מידע (במאמר מדובר בעיקר על Tor ו VPN).

המאמר מציע פתרון של גישה יעילה (מאוד) וחדשנית הנקראת FlowPics, המבוססת על המרת זרם נתונים לתמונה ולאחר מכן שימוש בטכניקות של עיבוד תמונה תוך שילוב עם למידה עמוקה CNN Convolutional neural network (בעברית - רשת נוירונים קונבנציונלית) על מנת לסווג את התעבורה.

כותבי המאמר ביצעו מחקר על מנת לבדוק כמה יעילה שיטת ניתוח התעבורה באמצעות גישה FlowPics, הם מתארים כיצד ניתן ליצור גרף ויזואלי שמייצג את גודל וכמות החבילות שהגיעו כתלות בזמן שבו אנו מסתכלים על תעבורת המידע באופן שיתן לנו להפוך אותו לממש תמונה ומדגישים כמה גדול ההבדל בין תמונה של תעבורה מוצפנת לתעבורה רגילה, וכמה שונות יכולות להיראות תעבורות של אפליקציות שונות, מסקנת החוקרים היא ששיטות סיווג המבוססות על מאפיינים ידניים (כמו סטטיסטיקות בסיסיות) לא תמיד מצליחות ללכוד את כל הדפוסים המורכבים הללו. לכן, הם עוברים לגישה של המרת כל הזרימה לתמונה (FlowPic) ושימוש ב CNN ש"ילמד" את הדפוסים הללו אוטומטית.

המאמר תורם לתחום הסיווג של תעבורת אינטרנט בכך שהוא מציג פתרון יעיל וחדשני לאתגרים הקיימים בסיווג תעבורה מוצפנת. הגישה המוצעת יכולה לשמש כבסיס לפיתוחים עתידיים בתחום ניהול רשתות ואבטחת מידע.

להלן כמה יתרונות עיקריים של הגישה המוצגים במאמר:

1. דיוק גבוה בסיווג תעבורה מוצפנת: השיטה מצליחה לסווג בדיוק גבוה תעבורה מוצפנת, כולל תעבורה שעוברת דרך VPN ו-Tor, שהיא מאתגרת מאוד לסיווג בשיטות מסורתיות.
2. זיהוי יישומים חדשים: הגישה מאפשרת זיהוי יישומים חדשים שלא היו חלק משלב האימון, מה שמראה על יכולת הכללה טובה.
3. שמירה על פרטיות: השיטה לא מסתמכת על תוכן המטען של המנות, ולכן לא פוגעת בפרטיות של המשתמשים.
4. יעילות חישובית: הגישה מאפשרת לעבוד עם חלון זמן קצר של זרימה חד-כיוונית, מה שמקטין את הצורך בחישובים מורכבים ומשפר את יעילות השיטה.

## מאפייני התעבורה בהם המאמר משתמש ואלו מהם מחדשים משהו?

### 1. גודל החבילות (Packet Size)

מה זה?

כמות הנתונים (בביתים) שנשלחים בכל מנה בודדת במהלך החיבור.

דוגמה:

בעת שידור וידאו, המנות יהיו גדולות יחסית ויכילו הרבה נתונים. בשונה מזה, הודעת צ'אט פשוטה תכיל מנות קטנות בהרבה.

שימוש:

המאפיין הזה נפוץ בניתוח תעבורת רשת ונועד לזהות את סוג השירות (וידאו, הורדה, גלישה).

החידוש במאמר:

השימוש במאפיין זה כחלק מייצוג היסטוגרמה דו-ממדית הממירה את גודל המנות לתמונה. בציר ה-X של FlowPic (הייצוג הוויזואלי) גודל המנות מוצג בטווח של 0-1500 בתים.

### 2. זמן הגעת המנות (Packet Arrival Time)

מה זה?

הזמן החולף בין הגעת שתי מנות עוקבות בפרק זמן מסוים.

דוגמה:

בחיבור לאיזשהו שירות של סטרימינג (כמו YouTube) נראה מנות המגיעות בפרקי זמן אחידים, בעוד שבתעבורת גלישה באינטרנט נראה זמני הגעה משתנים.

שימוש:

משמש בניתוח תעבורה כדי לזהות דפוסים של שירותים שונים (סטרימינג, הורדה, גלישה).

החידוש במאמר:

שילוב זמן ההגעה עם גודל החבילה בייצוג דו-ממדי חדשני. בציר ה-Y של FlowPic זמן ההגעה מוצג בדלתא של 0.01 שניות.

### 3. תדירות המנות (Packet Frequency)

מה זה?

מספר המנות שנשלחות או מתקבלות בפרק זמן מסוים (מנות לשנייה).

דוגמה:

חיבורי סטרימינג שולחים מנות בתדירות גבוהה וקבועה, בעוד שמערכות גיבוי שולחות מנות בקצב איטי יותר.

שימוש:

שימש בעבר לזיהוי התקפות (כגון DDoS) או להבדיל בין סוגי שירותים שונים.

החידוש במאמר:

המרת תדירות המנות לתמונה באמצעות יצירת FlowPic. בכל פיקסל בתמונה מופיעה עוצמת תדירות המנות בפרק זמן מסוים.

## החידוש העיקרי במאמר:

### 1. ייצוג ויזואלי (FlowPic)

#### מה זה?

המאמר מציע שיטה חדשנית הממירה נתוני תעבורה להיסטוגרמה דו-ממדית שמייצגת את דפוסי המנות בתמונה בגווני אפור.

#### למה זה חידוש?

מאפשר להפעיל רשתות נוירונים קונבולוציוניות (CNN), טכנולוגיה מתקדמת מעולם עיבוד התמונה, לסיווג תעבורה מוצפנת.

### 2. שימוש בחלון זמן קצר

#### מה זה?

המאמר משתמש בחלון זמן של 60 שניות בלבד כדי לאסוף ולנתח נתונים.

#### למה זה חידוש?

מאפשר ביצוע סיווג מהיר וחסכוני במשאבים לעומת שיטות אחרות המשתמשות בפרקי זמן ארוכים יותר.

### 3. אי-שימוש בתוכן המטען (Payload)

#### מה זה?

המאמר לא מנתח את תוכן המנות עצמן אלא רק את המאפיינים החיצוניים שלהן.

#### למה זה חידוש?

מאפשר ניתוח תעבורה מוצפנת מבלי לפגוע בפרטיות המשתמש, מה שהופך את השיטה ליעילה יותר מבחינה אתית ובטיחותית.

## לסיכום:

החדשנות המרכזית במאמר אינה במאפייני התעבורה עצמם, אלא באופן שבו הם משולבים יחד ומיוצגים בצורה ויזואלית. השיטה החדשה משתמשת בטכניקות עיבוד תמונה ולמידה עמוקה, מה שמוביל לשיפור משמעותי בדיוק הסיווג של סוגי התעבורה ברשת.

## תוצאות המרכזיות ומסקנות מתוך המאמר:

1. השוואה לשיטות קודמות (טבלה 3):

הטבלה מראה את דיוק הסיווג הממוצע שהושג על ידי שיטות שונות על מערך הנתונים ISCX VPN-nonVPN.

כפי שמצוין במאמר, קשה להשוות ישירות בין התוצאות עקב הבדלים בהגדרות הבעיה ובקטגוריות התעבורה.

למרות זאת, הטבלה מראה שהגישה המוצעת (FlowPic + CNN) משיגה דיוק סיווג ממוצע גבוה יותר מרוב השיטות האחרות, במיוחד אלו שלא משתמשות בנתוני מטען (payload).

Problem	FlowPic Acc. (%)	Best Previous Result	Remark
Non-VPN Traffic Categorization	85.0	84.0 % Pr., Gil <i>et al.</i> [15]	Different categories. [15] used unbalanced dataset
VPN Traffic Categorization	98.4	98.6 % Acc., Wang <i>et al.</i> [7]	[7] Classify raw packets data. Not including browsing category
Tor Traffic Categorization	67.8	84.3 % Pr., Gil <i>et al.</i> [15]	Different categories. [15] used unbalanced dataset
Non-VPN Class vs. All	97.0 (Average)	No previous results	
VPN Class vs. All	99.7 (Average)	No previous results	
Tor Class vs. All	85.7 (Average)	No previous results	
Encryption Techniques	88.4	99. % Acc., Wang <i>et al.</i> [7]	[7] Classify raw packets data, not including Tor category
Applications Identification	99.7	93.9 % Acc., Yamanavascular <i>et al.</i> [10]	Different classes

2. דיוק סיווג גבוה לקטגוריות תעבורה (טבלה 4):

דיוק הסיווג עבור קטגוריות התעבורה (VoIP, וידאו, העברת קבצים, צ'אט, גלישה) הוא גבוה מאוד, במיוחד עבור תעבורת Non-VPN ו-VPN.

עבור Non-VPN, הדיוק נע בין 93.3% (צ'אט) ל-99.6% (וידאו).

עבור VPN, הדיוק נע בין 83.6% (צ'אט) ל-99.9% (וידאו).

סיווג תעבורת Tor מציג דיוק נמוך יותר, אך עדיין סביר (בין 57.2% לגלישה ל-90.6% לווידיאו).

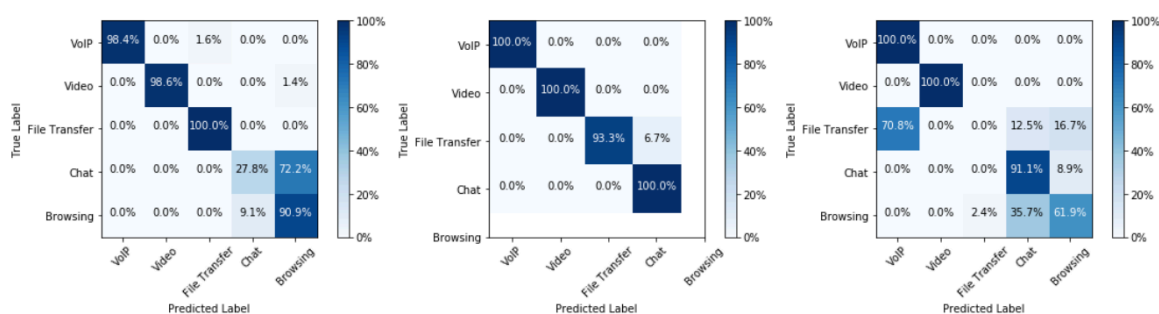
Class	Accuracy (%)			
VoIP	Training/Test	Non-VPN	VPN	Tor
	Non-VPN	<b>99.6</b>	99.4	48.2
	VPN	95.8	<b>99.9</b>	58.1
	Tor	52.1	35.8	<b>93.3</b>
Video	Training/Test	Non-VPN	VPN	Tor
	Non-VPN	<b>99.9</b>	98.8	83.8
	VPN	54.0	<b>99.9</b>	57.8
	Tor	55.3	86.1	<b>99.9</b>
File Transfer	Training/Test	Non-VPN	VPN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

3. מטריצות בלבול לסיווג קטגוריות (איור 4):

איור 4 מציג מטריצות בלבול עבור סיווג קטגוריות התעבורה עבור VPN, Non-VPN ו-Tor.

מטריצות אלו מאפשרות לראות אילו קטגוריות מתבלבלות זו בזו.

ניתן לראות שהבלבול העיקרי בתעבורת Non-VPN ו-VPN הוא בין צ'אט לגלישה, בעוד שבתעבורת Tor הבלבול גדול יותר בין כל הקטגוריות.



(a) Non-VPN

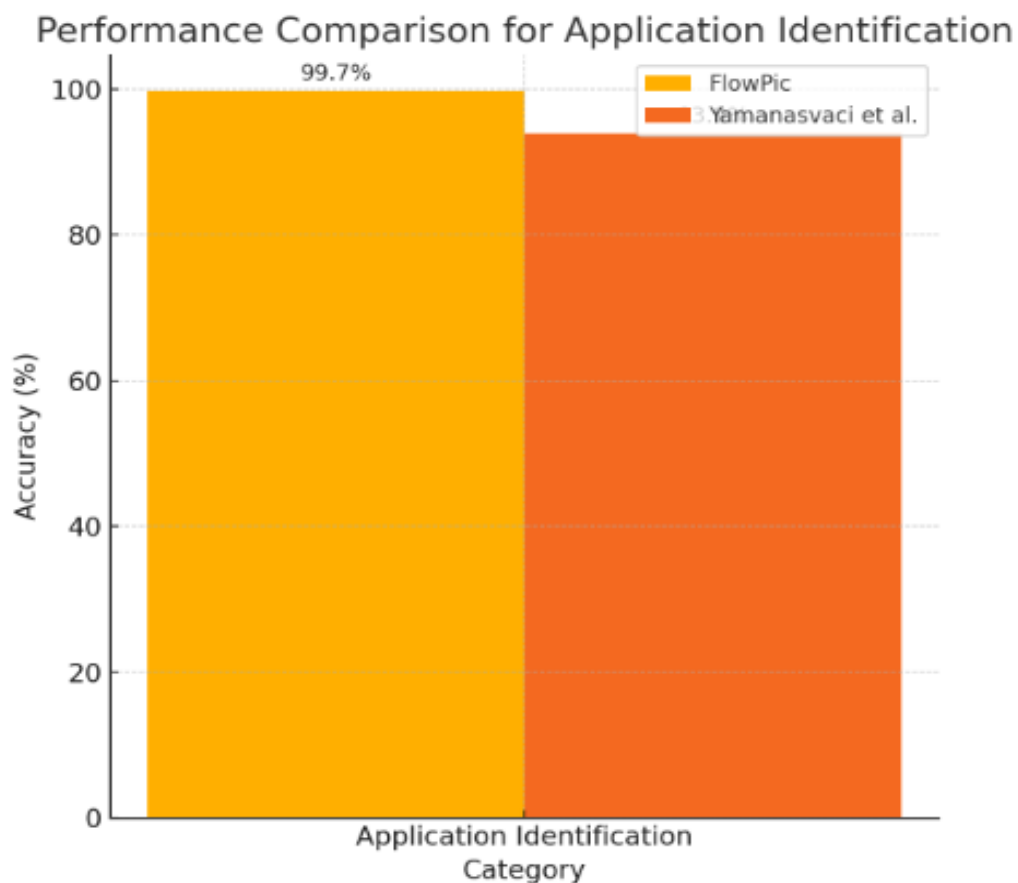
(b) VPN

(c) Tor

## מסקנות עיקריות:

1. הגישה משפרת את ביצועי הסיווג בהשוואה לשיטות קודמות:  
טבלה III מספקת אינדיקציה לכך שהגישה המוצעת משפרת את ביצועי הסיווג בהשוואה לשיטות קודמות, במיוחד אלו שלא משתמשות בנתוני מטען.
2. הגישה יעילה לסיווג תעבורת רשת מוצפנת (במיוחד VPN):  
הדיוק הגבוה בסיווג תעבורת VPN (טבלה IV) מצביע על כך שהגישה יעילה במיוחד לסיווג תעבורה מוצפנת, וזהו יתרון משמעותי.
3. סיווג תעבורת Tor מאתגר יותר:  
הדיוק הנמוך יותר בסיווג תעבורת Tor מצביע על כך שסוג זה של תעבורה קשה יותר לסיווג, כנראה בגלל המאפיינים הייחודיים של Tor (שכבות הצפנה מרובות, שינוי נתיבים תדיר).
4. קיימים הבדלים במאפיינים של קטגוריות תעבורה שונות:  
ההבדלים בדיוק הסיווג בין הקטגוריות השונות (טבלה IV) מצביעים על כך שלכל קטגוריה יש מאפיינים ייחודיים המקלים או מקשים על הסיווג. לדוגמה, תעבורת וידאו קלה יותר לסיווג מתעבורת צ'אט.

על מנת להמחיש את התוצאות בצורה מיטבית יצרנו גרף שמשווה את תוצאות הניסוי של סיווג התעבורה של FlowPic לעומת המחקרים שעשו UNB Group ו Wang et al , ובנוסף , גרף המשווה בין זיהוי האפליקציות בהן השתמשו במחקר לעומת המחקר השני שאליו בוצעה ההשוואה:





## מאמר מספר 2 :

"Early Traffic Classification With Encrypted ClientHello A Multi Country Study"

### מהי התרומה המרכזית של המאמר?

המאמר עוסק בסיווג תעבורה מוקדם בסביבת Encrypted ClientHello, המקשה על זיהוי סוג התעבורה בשל הצפנת מידע רגיש ב-TLS handshake. כתרומה מרכזית, המאמר מציג אלגוריתם חדש בשם hRFTC (hybrid Random Forest Traffic Classifier), המשלב ניתוח של מטא-דאטה לא מוצפנת ב-TLS handshake עם מאפיינים סטטיסטיים של זרימות התעבורה, כמו גודל מנות וסדרות זמן מבוססות זרימה. בנוסף, המאמר מציג מערך נתונים מקיף ועדכני שאסוף ממדינות שונות בצפון אמריקה, אירופה ואסיה, הכולל למעלה מ-600,000 זרימות TLS המחולקות ל-19 סוגי תעבורה שונים, ומאפשר מחקר מעמיק של ביצועי אלגוריתמים שונים בסביבת ECH. המאמר מנתח את היכולת של אלגוריתמים קיימים להתמודד עם אתגרי ה-ECH ומדגים את החשיבות של התאמת מודלים של סיווג תעבורה למיקומים גיאוגרפיים שונים עקב הבדלים בדפוסי התעבורה. מחקר זה מראה כי אלגוריתם hRFTC עולה בביצועיו על אלגוריתמים חדישים אחרים, ומספק פתרון יעיל לסיווג תעבורה מוקדם בסביבות רשת מודרניות.

### באילו מאפייני תעבורה המאמר משתמש, ואילו מהם הם חדשים?

סיווג תעבורה (TC-TRAFFIC CLASSIFICATION):

הוא תהליך לזהות וסיווג של תעבורת רשת על פי מאפיינים שונים ומטרתו היא לשפר את איכות השירות ברשתות על ידי התאמה אוטומטית של משאבים כמו רוחב פס.

סיווג תעבורה (Transport Layer Security) TLS:

בתחילת תהליך ה-handshake, הלקוח שולח לשרת הודעת ClientHello, והשרת משיב בהודעת ServerHello. ההודעה ClientHello מכילה מידע רגיש, כולל Server Name Indication (SNI), אשר חושף את שם הדומיין אליו הלקוח מנסה להתחבר.

בגרסה TLS 1.3, כל השלבים הקריטיים בתהליך ה-handshake מוצפנים, וכן התקשורת המאובטחת בין הצדדים. לאחר השלמת ה-handshake, כל המידע המועבר בין הלקוח לשרת מוצפן, כך שרק הצדדים המשתתפים בתקשורת יכולים לגשת אליו.

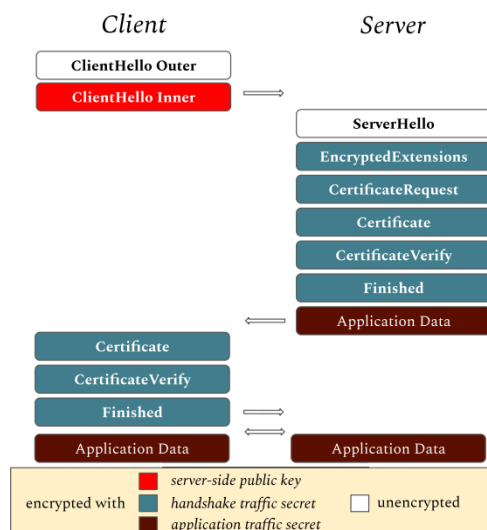


FIGURE 1. TLS 1.3 handshake with encrypted ClientHello.

## ECH:

ECH מהווה שיפור משמעותי ל-TLS 1.3, הוא מגן על פרטיות המשתמשים בשלב קריטי של הקמת החיבור המאובטח (SNI).

יש כמה תרומות מרכזיות של המאמר על ECH:

1. יצירת מאגר נתונים מהגדולים והמעודכנים ביותר לסיווג מוקדם של תעבורת TLS, הכולל מעל 600,000 זרמים מ-19 קטגוריות שונות שנאספו מאזורים גאוגרפיים מגוונים, ומספק תמונה ריאליסטית של תעבורת האינטרנט המודרנית.
2. פיתוח אלגוריתם חדש (hRFTC) - אלגוריתם היברידי חדש לסיווג תעבורה המשתמש במאפייני TLS לא מוצפנים ובמאפייני זרימה סטטיסטיים.
3. השוואת ביצועים - הוכחת עליונות האלגוריתם החדש על פני אלגוריתמים מתקדמים אחרים עם דיוק סיווג של עד 94.6%.
4. בדיקת הטרוגניות בנתונים - ניתוח כיצד אלגוריתמי סיווג מתמודדים עם שונות גאוגרפית ושירותים שונים.

### יש שני סוגי מאפיינים עיקריים לסיווג מוקדם של תעבורת רשת:

מאפיינים מבוססי חבילות (Packet-based Features): שילוב של מידע מתוך הודעות ClientHello ו-ServerHello שאינן מוצפנות, כולל גודל ותוכן של הודעות TLS, תבניות בהודעות ההתחלתיות והעדפות הצפנה, המאפשרות זיהוי ראשוני של סוג התעבורה.

מאפיינים מבוססי זרימה (Flow-based Features): שימוש בסטטיסטיקות של גדלי חבילות (Packet Size - PS), זמני הגעה בין חבילות (Inter-Packet Time - IPT), והבחנה בין תעבורת העלאה (uplink) להורדה (downlink), מה שמאפשר ניתוח עמוק של התנהגות התעבורה לאורך זמן.

המאמר מציע אלגוריתם חדש, (Hybrid Random Forest Traffic Classifier (hRFTC), המיועד לסיווג מוקדם של תעבורת רשת מוצפנת, במיוחד בתרחישים שבהם נעשה שימוש ב-*Encrypted ClientHello* (ECH)) בפרוטוקול TLS. האלגוריתם משלב בין מאפייני חבילות TLS שאינן מוצפנות לבין מאפיינים סטטיסטיים מבוססי זרימה, כגון גודל חבילות, זמני הגעה ופילוח תעבורת העלאה והורדה, במטרה לשפר את דיוק הסיווג גם כאשר מידע קריטי כמו SNI מוצפן.

החידושים המרכזיים של hRFTC כוללים:

- שימוש משולב במאפייני חבילות וזרימה: שילוב מידע מתוך ה-*TLS handshake* הבלתי מוצפן יחד עם נתוני זרימה סטטיסטיים משפר את איכות הסיווג גם במצבים בהם חלק גדול מהמידע מוסתר.
- תמיכה בתעבורה מבוססת QUIC: האלגוריתם מתרחב מעבר ל-TLS-over-TCP ותומך גם בפרוטוקול QUIC, באמצעות עיבוד המידע הבלתי מוצפן של TLS ושימוש בסטטיסטיקות זרימה מתקדמות.
- קריטריון חדש לבחירת חבילות לסיווג: במקום לנתח מספר קבוע של חבילות, האלגוריתם מבצע ניתוח עד לחבילה הראשונה המכילה נתוני אפליקציה, מה שמאפשר סיווג מהיר ומדויק יותר.

אלגוריתם hRFTC מבוסס על Random Forest, אשר מאפשר סיווג יעיל ומדויק של תעבורת רשת מוצפנת, גם כאשר מידע קריטי כמו SNI מוסתר באמצעות ECH. תוצאות המחקר מצביעות על כך שהאלגוריתם החדש משיג דיוק של עד 94.6%, מה שמציב אותו מעל האלגוריתמים המובילים כיום בתחום סיווג התעבורה המוצפנת.

## מהם הממצאים המרכזיים, ואילו תובנות עולות מהם?

1. **דיוק גבוה בסיווג התעבורה** – האלגוריתם החדש hRFTC משיג דיוק של עד 94.6% מה שמעלה אותו מעל האלגוריתמים המובילים הקיימים, במיוחד בתרחישים שבהם נעשה שימוש ב-ECH.
2. **שיפור משמעותי לעומת שיטות קיימות** – אלגוריתמים המבוססים רק על מאפייני TLS בלתי מוצפנים הגיעו לדיוק של 38.4% בלבד, בעוד שהשילוב של מאפייני חבילות זרימה ב-hRFTC שיפר משמעותית את ביצועי הסיווג.
3. **שונות גיאוגרפית** – נמצא כי אלגוריתמים המאומנים במדינה מסוימת אינם בהכרח יעילים במדינות אחרות, מה שמדגיש את הצורך באימון מחדש בהתאם למיקום הגיאוגרפי.
4. **תרומת מאפייני זרימה** – מאפייני זרימה, כמו גודל חבילות וזמני הגעה ביניהן, תרמו תרומה משמעותית לסיווג התעבורה, והוכחו כחשובים לא פחות ממאפייני ה-TLS.
5. **התמודדות עם הצפנת ECH** – למרות שהצפנת ה-ClientHello מקשה על הסיווג המסורתי, השיטה ההיברידית של hRFTC מצליחה להתגבר על האתגר באמצעות שילוב חכם של נתונים סטטיסטיים ומידע בלתי מוצפן.

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

## תובנות:

1. **השפעת ECH על סיווג תעבורה:**  
השימוש ב-ECH (*Encrypted ClientHello*) מצמצם משמעותית את יכולת הסיווג של שיטות מסורתיות, מאחר שהוא מצפין מידע קריטי כמו *SNI*. כתוצאה מכך, יש צורך בגישות היברידיות המשלבות מאפייני חבילות וזרימה כדי לשמר את דיוק הסיווג.
2. **תרומת מאפייני זרימה לסיווג מדויק:**  
פרמטרים כגון גודל חבילות, זמני הגעה והבדלים בין תעבורת העלאה (uplink) להורדה (downlink) מספקים מידע חיוני לזיהוי סוגי שירותים שונים גם בתעבורה מוצפנת.
3. **התמודדות עם QUIC:**  
השימוש בפרוטוקול QUIC מוסיף מורכבות בשל הצפנת ה-TLS handshake והיעדר מידע זמין כמו ב-TLS-over-TCP. עם זאת, שילוב סטטיסטיקות זרימה מתקדמות מאפשר עדיין סיווג מדויק של התעבורה.
4. **צורך באימון מותאם גיאוגרפי:**  
בשל הבדלים בתשתיות רשת, ספקי CDN, והגדרות שרתים במדינות שונות, נדרש לבצע אימון מחדש של המודל לכל אזור גיאוגרפי כדי לשפר את ביצועי הסיווג ולהתאים אותו לתנאי הרשת המקומיים.

### מאמר מספר 3 :

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

#### מהי התרומה המרכזית של המאמר?

התרומה המרכזית של המאמר היא הצגת גישה תקיפה 'פאסיבית' המנצלת נקודות תורפה באפליקציית HTTPS מוצפנת, ובכך מאפשרת זיהוי מדויק של מערכת ההפעלה, הדפדפן והאפליקציה שבהם המשתמש משתמש. המחברים מראים כי למרות השימוש בפרוטוקולי הגנה כמו SSL/TLS, ניתן לנצל את המאפיינים החיצוניים של התעבורה – הן באמצעות תכונות סטטיסטיות בסיסיות והן באמצעות תכונות חדשות המבוססות על דפוסי ההתנהגות המתפרצת (bursty) של הדפדפנים והתנהגות ה-SSL – כדי להגיע לדיוק סיווג גבוה (93.51% בתכונות בסיסיות, עד 96.06% בשילוב עם התכונות החדשות). בנוסף, המחקר מלווה ב-data set גדול שנאסף באופן אוטומטי, המאפשר אימון ובדיקה מדוקדקים של המודל.

כך, תרומתו המרכזית טמונה בכך שהמחקר מוכיח את האפשרות לזהות פרטי מערכת, דפדפן ואפליקציה למרות שהתעבורה מוצפנת, דבר המצביע על נקודת תורפה בשיטות ההצפנה הקיימות כיום.

#### באילו מאפייני תעבורה המאמר משתמש, ואילו מהם הם חדשים?

המאמר עושה שימוש בשתי קבוצות מאפיינים: קבוצת תכונות בסיסיות הקשורות לתעבורת הרשת, וקבוצת תכונות חדשות שהוצעו על ידי החוקרים לשיפור הדיוק. השילוב בין שתי הקבוצות מאפשר לזהות באופן מדויק את מערכת ההפעלה, סוג הדפדפן והאפליקציה של המשתמש, באמצעות קריאת הפקטות בתעבורת HTTPS.

התכונות החדשות בהם החוקרים הוסיפו לטובת דיוק הממצאים

TCP initial window size
TCP window scaling factor
# SSL compression methods
# SSL extension count
# SSL cipher 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
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 Maximum Segment Size
Forward SSL Version

(b) new features

התכונות הבסיסיות משמשות להרבה סיווגי תעבורה

# 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

(a) base features

## מהם הממצאים המרכזיים, ואילו תובנות עולות מהם?

**הממצאים** מסתמכים על data set שנאגר ע"י אוטומצית רשת בשם 'סלניום' אשר מגיעה עם crawlers (תוכנה הסורקת אוטומטית דף אחר דף, אינדקסים וקישורים) וכל התעבורה הוקלט דרך פורט 443 (TLS\SSL) ולאחר מכן חילקו את התעבורה לקטעים של פעילויות. ה data set מכיל יותר מ 20 אלף פעילויות.

לטובת האנליזה השתמשו בבינה מלכותית מבוקרת אשר מקבלת דוגמית של פעילות מסוימת ומחזירה קטלוג של פעילות ע"י טאפל. למידת המכונה התחלקה ל- 70 אחוז אימון ו30 אחוז בדיקות.

הממצאים מוצגים בתור "מטריצת בלבול" אשר מציגה בציר ה Y את הפעילות האמיתית שמתקיימת אשר מתוויגת בתור <OS, Browser, Application> ובציר ה X מופיעות פעילויות שהאלגוריתם ניבא באותן קטגוריות. כל תא בטבלה מציג את ההסתברות שבה האלגוריתם הצליח לזהות נכון את סוג הפעילות בהתבסס על הפרמטרים והתכונות שנבחרו לחילוף המידע.

		Predicted labels																															
		Windows IExplorer Twitter	Ubuntu Firefox Google-Background	Windows Non-Browser Microsoft-Background	Windows Chrome Twitter	Windows Firefox Twitter	OSX Safari Google-Background	OSX Safari Youtube	Ubuntu Chrome Unknown	Windows Chrome Google-Background	Ubuntu Firefox Twitter	OSX Safari Unknown	Ubuntu Firefox Unknown	Ubuntu Chrome Google-Background	Ubuntu Chrome Twitter	Windows Firefox Google-Background	OSX Safari Twitter	Ubuntu Firefox Youtube	Windows Non-Browser Teamviewer	Ubuntu Chrome Youtube	Windows Non-Browser Dropbox	Windows Chrome Unknown	Ubuntu Chrome Facebook	Windows Firefox Unknown	Ubuntu Firefox Facebook	OSX Chrome Twitter	Windows Explorer Unknown	Ubuntu Non-Browser Microsoft-Background	Windows IExplorer Google-Background	OSX Chrome Google-Background	OSX Chrome Unknown		
Real labels	Windows IExplorer Twitter	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
	Ubuntu Firefox Google-Background	0	.97	0	0	0	0	0	0	0	0	0	.01	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
	Windows Non-Browser Microsoft-Background	0	0	.99	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	Windows Chrome Twitter	0	0	0	.99	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	.01	0	0	0	0	0	0	0	0	0	0	
	Windows Firefox Twitter	0	0	0	0	.98	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	.02	0	0	0	0	0	0	0	0	0	
	OSX Safari Google-Background	0	0	0	0	0	.92	.04	0	0	.02	0	0	0	0	0	.02	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	OSX Safari Youtube	0	0	0	0	0	.02	.97	.01	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	Ubuntu Chrome Unknown	0	0	0	0	0	0	0	.84	0	0	0	0	.07	.04	0	0	0	0	.01	0	0	.03	0	0	0	0	0	0	0	0	0	
	Windows Chrome Google-Background	0	0	.01	.03	0	0	0	0	.94	0	0	0	0	0	0	.02	0	0	0	0	.01	0	0	0	0	0	0	0	0	0	0	0
	Ubuntu Firefox Twitter	0	0	0	0	0	0	0	0	0	.95	0	.03	0	0	0	0	0	.01	0	0	0	0	0	0	0	0	0	0	0	0	0	
	OSX Safari Unknown	0	0	0	0	0	0	.06	.01	0	0	0	.91	0	0	0	.01	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	Ubuntu Firefox Unknown	0	.02	0	0	0	0	0	0	0	0	.08	0	.87	0	0	0	0	.01	0	0	0	0	0	.03	0	0	0	0	0	0	0	
	Ubuntu Chrome Google-Background	0	.07	0	0	0	0	0	.18	0	0	0	0	0	.73	0	0	0	0	.02	0	0	0	0	0	0	0	0	0	0	0	0	
	Ubuntu Chrome Twitter	0	.02	0	0	0	0	0	.08	0	0	0	0	.03	.84	0	0	0	.01	0	0	0	.01	0	.08	0	0	0	0	0	0	0	0
	Windows Firefox Google-Background	0	0	0	.01	0	0	0	0	.01	0	0	0	0	0	0	.97	0	0	0	0	0	0	.01	0	0	0	0	0	0	0	0	0
	OSX Safari Twitter	0	0	0	0	0	0	.06	0	0	0	.03	0	0	0	0	0	.91	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Ubuntu Firefox Youtube	0	.02	0	0	0	0	0	0	0	.02	0	.02	0	0	0	0	0	.93	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Windows Non-Browser Teamviewer	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
	Ubuntu Chrome Youtube	0	0	0	0	0	0	0	.07	0	0	0	0	0	.13	.04	0	0	0	0	.74	0	.02	0	0	0	0	0	0	0	0	0	0
	Windows Non-Browser Dropbox	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
Windows Chrome Unknown	0	0	.02	.09	0	0	0	0	.02	0	0	0	0	0	0	0	0	0	0	0	0	.86	0	0	0	0	0	0	0	0	0	0	0
Ubuntu Chrome Facebook	0	0	0	0	0	0	0	.3	0	0	0	0	.04	0	0	0	0	0	0	0	0	.67	0	0	0	0	0	0	0	0	0	0	
Windows Firefox Unknown	0	0	.06	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	.94	0	0	0	0	0	0	0	0	0	
Ubuntu Firefox Facebook	0	.06	0	0	0	0	0	0	0	.11	0	.28	0	0	0	0	0	0	0	0	0	0	0	.56	0	0	0	0	0	0	0	0	
OSX Chrome Twitter	0	0	0	0	0	0	0	.13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	.75	0	0	0	.06	.06	0	0	
Windows IExplorer Unknown	.71	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	.29	0	0	0	0	0	0	0	
Ubuntu Non-Browser Microsoft-Background	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Windows IExplorer Google-Background	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
OSX Chrome Google-Background	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
OSX Chrome Unknown	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

(a) Tuple Confusion Matrix

Real labels	Predicted labels		
	Windows	Ubuntu	OSX
Windows	1	0	0
Ubuntu	0	1	0
OSX	0	0	1

(b) OS Confusion Matrix

Real labels	Predicted labels				
	Chrome	Firefox	IExplorer	Safari	Non-Browser
Chrome	.97	.02	0	0	0
Firefox	.01	.98	0	0	0
IExplorer	0	0	1	0	0
Safari	.01	0	0	.99	0
Non-Browser	.03	0	0	0	.96

(c) Browser Confusion Matrix

Real labels	Predicted labels							
	Dropbox	Facebook	Google-background	Microsoft-Background	Teamviewer	Twitter	Youtube	Unknown
Dropbox	.98	0	.02	0	0	0	0	0
Facebook	0	.62	.04	0	0	.04	0	.29
Google-background	0	0	.95	0	0	.01	.01	.03
Microsoft-Background	0	0	0	.96	0	0	0	.04
Teamviewer	0	0	0	0	1	0	0	0
Twitter	0	0	0	0	0	.98	0	.01
Youtube	0	0	.03	0	0	.02	.93	.01
Unknown	0	.02	.04	.01	0	.05	.01	.86

(d) Application Confusion Matrix

## **תובנות:**

הסיווג לרוב הפעילויות היה כמעט מושלם, אם דיוק של כמעט 1 לרבות טופלים עם שם זהה וגם טאפלים שלא ניתן לזהות. סיווג מערכות ההפעלה הוא מושלם ללא טעויות, סיווג סוגי הדפדפנים הוא כמעט מושלם, אך סיווג סוגי האפליקציות גם כמעט מושלם למעט פייסבוק, האלגוריתם טעה בו ב-29 אחוז מזמן הלמידה וקיטלג אותו בתור אפליקציה לא ידועה.

לסיכום, ניתן לזהות תעבורה מוצפנת ולהוציא מהם מידע כגון איזה סוג מערכת הפעלה, דפדפן ואפליקציה המשתמש משתמש דרך המחשב הנייח או הנייד שלו. ניתן לראות, למרות שימוש בפרוטוקולי הגנה כגון SSL/TLS, שניצול נקודות התורפה באפליקציית ה-HTTPS מאפשר לתוקף לזהות דפוסים התנהגותיים ולהסיק מידע רגיש אודות המשתמש, מה שעלול לסכן את פרטיותו ואף לאפשר מתקפות ממוקדות.

## חלק 3: ניתוח תעבורת רשת וזיהוי יישומים באמצעות Wireshark: אפיון יישומים וסימולציית תקיפה

1. **הקלטת תעבורת רשת ואיסוף נתונים**
  - הגדרת מתודולוגיית הניסוי והגבלת "רעש" תעבורתי
  - אפיון סוגי היישומים הנבדקים
  - שמירת מפתחות TLS לצורך ניתוח נתונים מוצפנים
2. **ניתוח והשוואת מאפייני התעבורה בין יישומים**
  - מאפייני התעבורה הנבדקים (שדות כותרת IP, שדות כותרת TCP, שדות TLS, גודל חבילות, זמני הגעה, נפח התעבורה)
  - הצגת גרפים וניתוח השוואתי של סוגי התעבורה
  - דפוסי שימוש ייחודיים לכל יישום
3. **הסבר מפורש של הגרפים והממצאים**
  - השוואת גודל חבילות בין היישומים
  - השפעת זמני ההגעה בין חבילות על זיהוי סוג השירות
  - זיהוי דפוסי תעבורה ייחודיים
4. **סימולציה של תקיפה לזיהוי אפליקציות על בסיס תעבורה מוצפנת**
  - תרחיש 1: התוקף מכיר את גודל החבילות, חותמות זמן, ו-*hash* של מזהה הזרימה
  - תרחיש 2: התוקף מכיר רק את גודל החבילות וחותמות הזמן
  - ניתוח היכולת של התוקף לזהות את האפליקציה לפי הנתונים הזמינים
  - שיטות למזעור הסיכון ודרכים למניעת זיהוי
5. **ניסוי בונוס: תעבורת רקע והשפעת שימוש משולב ביישומים**
  - השפעת יישומים משניים על ניתוח תעבורה (למשל, גלישה ברקע תוך כדי האזנה לספוטיפיי)
  - השפעת תעבורה מעורבת על יכולת הזיהוי של התוקף
  - הצעת ניסויים חלופיים והשלכותיהם

## 3.1. הקלטת תעבורת רשת ואיסוף נתונים

### תיאור תהליך ההקלטה והלכידה של התעבורה

במהלך העבודה, השתמשנו בתוכנת **Wireshark** כדי ללכוד ולנתח את תעבורת הרשת של מספר אפליקציות נפוצות בהתאם להנחיות. לצורך הבדיקה, כל אפליקציה נבדקה בנפרד על מנת למנוע "רעש" (תעבורה לא רצויה) ולאפשר ניתוח מדויק של תכונות התעבורה. בנוסף, מפתחות ההצפנה נשמרו מראש כדי לאפשר פענוח של התעבורה המוצפנת לאחר מכן.

### אנו מדגישים כי ביטלנו את פרוטוקול QUIC על מנת שנוכל לפתוח את ההצפנה של המפתחות בצורה מיטבית ולכן אנו מתייחסים לתוצאות בצורה כזו שהוא לא מופיע!

פירוט על כל אחת מההקלטות השונות:

#### 1. הקלטת תעבורת גלישה בדפדפן Chrome

- נכנסנו לאתר **Mako** לרשימת הפרקים של הסדרה **הפרלמנט** דרך דפדפן **Google Chrome**.
- המתנו לטעינה מלאה של העמוד כדי להבטיח שכל הבקשות נטענו במלואן.
- לאחר סיום הטעינה, עצרנו את ההקלטה ושמרנו את מפתחות ההצפנה (SSL/TLS) על מנת לפענח את המידע המוצפן בניתוח מאוחר יותר.

#### 2. הקלטת תעבורת גלישה בדפדפן Firefox

- חזרנו על התהליך עם אותו עמוד באתר **Mako**, הפעם באמצעות דפדפן **Mozilla Firefox**.
- שוב, המתנו לטעינה מלאה לפני עצירת ההקלטה ושמרנו את מפתחות ההצפנה לצורך פענוח.

#### 3. הקלטת תעבורת הזרמת שמע (Audio Streaming) ב-Spotify

- נכנסנו ל-Spotify והפעלנו הזרמת שמע.
- על מנת לנתח את הבקשות המוצפנות, ביטלנו את פרוטוקול **QUIC** המוגדר כברירת מחדל.
- פעולה זו אילצה את התקשורת לעבור דרך פרוטוקול **TCP**, מה שאפשר לתפוס את הבקשות באמצעות Wireshark ולפענח את הנתונים המוצפנים בעזרת מפתחות ההצפנה.

#### 4. הקלטת תעבורת הזרמת וידאו (Video Streaming) ב-YouTube

- גלשנו ל-YouTube דרך דפדפן **Google Chrome** והפעלנו סרטון וידאו.
- במהלך ההקלטה סיננו את התעבורה כך שתוצג רק התעבורה הרלוונטית לשרת הוידאו שמבצע את הסטרימינג.
- פעולה זו אפשרה לנתח את תכונות התעבורה הרלוונטיות בלבד, כמו גודל הפקטות והפרוטוקולים המעורבים.

#### 5. הקלטת תעבורת שיחת וידאו (Video Conference) ב-Zoom

- פתחנו פגישה ב-Zoom והוספנו מספר משתתפים נוספים.
- במהלך הפגישה, דיברנו במשך מספר שניות כדי לייצר תעבורת וידאו ושמע אמיתית.
- עם סיום השיחה, סגרנו את הפגישה ואת ההקלטה במקביל.



## 3.2. ניתוח והשוואת מאפייני התעבורה בין יישומים

על מנת ליצור את הגרפים השתמשנו בחבילות הבאות שעל הבודק להתקין כאשר הוא מריץ את הקוד:

### :Matplotlib

#### למה זה משמש?

ספריית גרפים שמאפשרת ליצור גרפים דו-ממדיים במגוון פורמטים – עמודות, עוגות, קווי מגמה ועוד.

#### שימוש עיקרי בפרויקט:

הצגת השוואות בין אפליקציות שונות (לפי פרוטוקולים, גודל פקטות וכו').  
הצגת תיאורים ויזואליים של התעבורה ברשת.

### :Pyshark

#### למה זה משמש :

ספרייה לניתוח קבצי תעבורת רשת , המאפשרת גישה נוחה לניתוח חבילות.

#### שימוש עיקרי בפרויקט :

קריאה וניתוח של קבצי הקלטה של ה - Wireshark.  
חילוץ מידע מהחבילות כמו כתובות IP, פרוטוקולים, פורטים ונתוני TLS להשוואה בין אפליקציות.

### :Pandas

#### למה זה משמש :

ספרייה לניהול וניתוח נתונים בטבלאות.  
גם מכילה כלים מתקדמים לסינון, קיבוץ וניתוח סטטיסטי של נתונים.

#### שימוש עיקרי בפרויקט :

לשמור את הנתונים שהופקו מקבצי ה- PCAP בטבלאות מסודרות.  
ניתוח נתונים , הכנה להצגה גרפית של התוצאות.

### :Os

#### למה זה משמש :

ספרייה מובנית ב Python המאפשרת אינטראקציה עם מערכת ההפעלה.  
משמשת בעיקר לניהול קבצים ותיקיות, גישה לנתיבי קבצים ועוד.

#### שימוש עיקרי בפרויקט :

איתור כל קבצי ההקלטה בתיקיה אחת.  
טעינה אוטומטית של הקבצים לניתוח ללא צורך בהזנה ידנית.  
עבודה דינמית עם נתיבים עבור פונקציית הניתוח.

### :Numpy

#### למה זה משמש?

ספריית Python שמיועדת לעבודה עם מערכים, וקטורים ופעולות מתמטיות מתקדמות.

#### שימוש עיקרי בפרויקט:

יצירת מערכים לניהול נתונים גרפיים (למשל, מיקומים של עמודות בגרפים).  
חישובים מתמטיים כמו ממוצעים, סטיית תקן ועוד.

### 3.3. הסבר מפורט של הגרפים והממצאים

#### פירוט גרף A: התפלגות פרוטוקולי IP

##### מה מייצג הגרף?

הגרף מציג את התפלגות הפרוטוקולים שנמצאו בתעבורת הרשת כנגד כמות החבילות שהגיעה בחמש אפליקציות שונות (Web-surfing 1, Web-surfing 2, Audio Streaming, Video Streaming, Video Conference).

##### ציר ה-X (פרוטוקול):

מציג את סוגי הפרוטוקולים ששימשו להעברת נתונים בין השרת ללקוח. הפרוטוקולים המוצגים כוללים:

**UDP:** פרוטוקול מהיר אך לא אמין (ללא תיקוני שגיאות), נפוץ לשיחות וידאו ואודיו

**DATA:** תעבורת נתונים גולמית (בדרך כלל עבור שירותי סטרימינג או שיחות וידאו).

**DNS:** בקשות לתרגום כתובות דומיין לכתובות IP.

**(DTLS) Datagram Transport Layer Security:** גרסה של TLS מעל UDP, משמש לאבטחת תעבורת UDP (למשל ב-Zoom).

**(STUN) Session Traversal Utilities for NAT:** פרוטוקול לזיהוי כתובת IP חיצונית עבור חיבורים.

**(TCP) Transmission Control Protocol:** פרוטוקול אמין לתקשורת שמבצע בדיקות שגיאה.

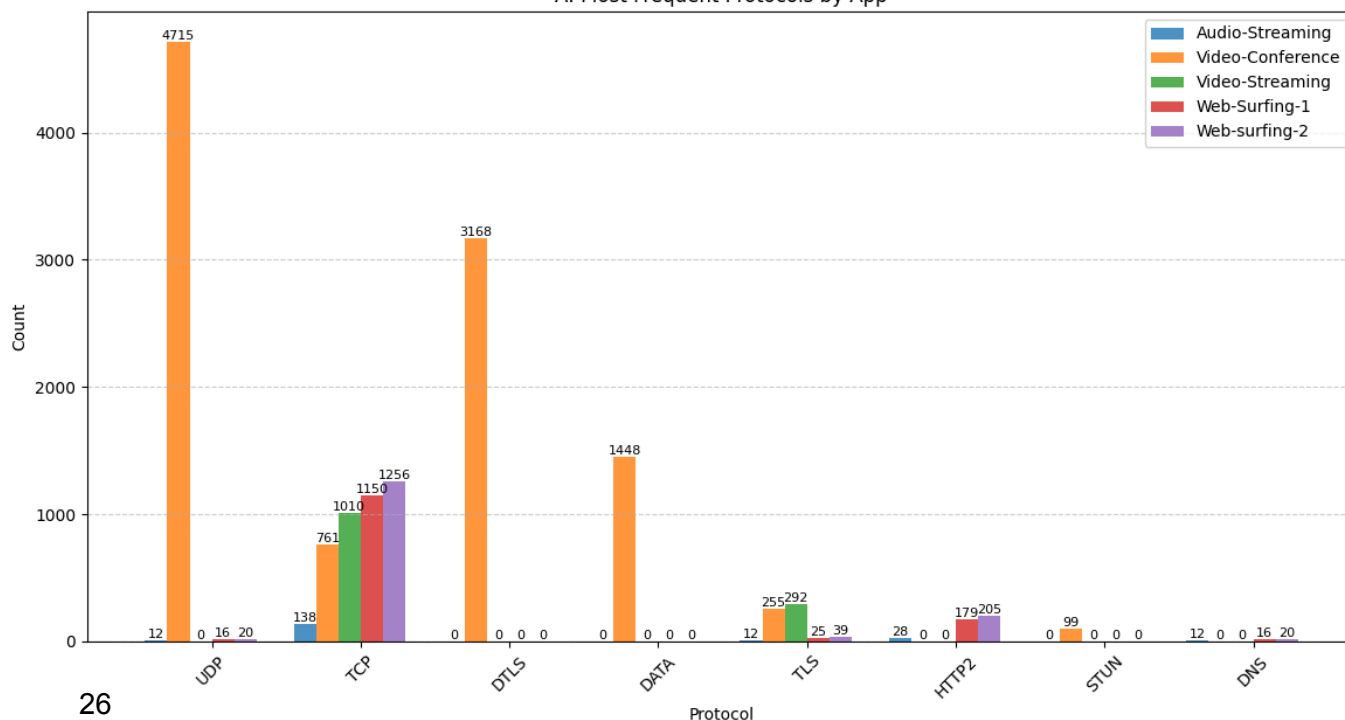
**(TLS) Transport Layer Security:** שכבת אבטחה על גבי TCP.

##### ציר ה-Y (מספר החבילות):

מציין את מספר החבילות שנתפסו עבור כל פרוטוקול, בכל אפליקציה.

משקף את רמת השימוש של כל פרוטוקול בכל אחת מהאפליקציות.

A: Most Frequent Protocols by App



## פירוט גרף B : עשרת הפורטים הנפוצים ביותר ב-TCP:

### מה מייצג הגרף?

הגרף מציג את 10 הפורטים הנפוצים ביותר בפרוטוקול TCP כנגד כמות החבילות שהגיעה בתעבורת הרשת שנלכדה מחמש אפליקציות שונות:

**Web-surfing 1, Web-surfing 2, Audio Streaming, Video Streaming, Video Conference**

### ציר ה-X (פורט המקור):

מציג את מספרי הפורטים ששימשו כנקודת התחלה (Source Port) במחשב הלקוח לשליחת בקשות אל השרת.

כל מספר פורט מייצג "דלת" בתקשורת, שדרכה נשלחים ונקלטים נתונים.

### המשמעות של הפורטים הנפוצים:

**443:** פורט ברירת מחדל עבור תעבורת HTTPS המאובטחת (TCP/TLS).

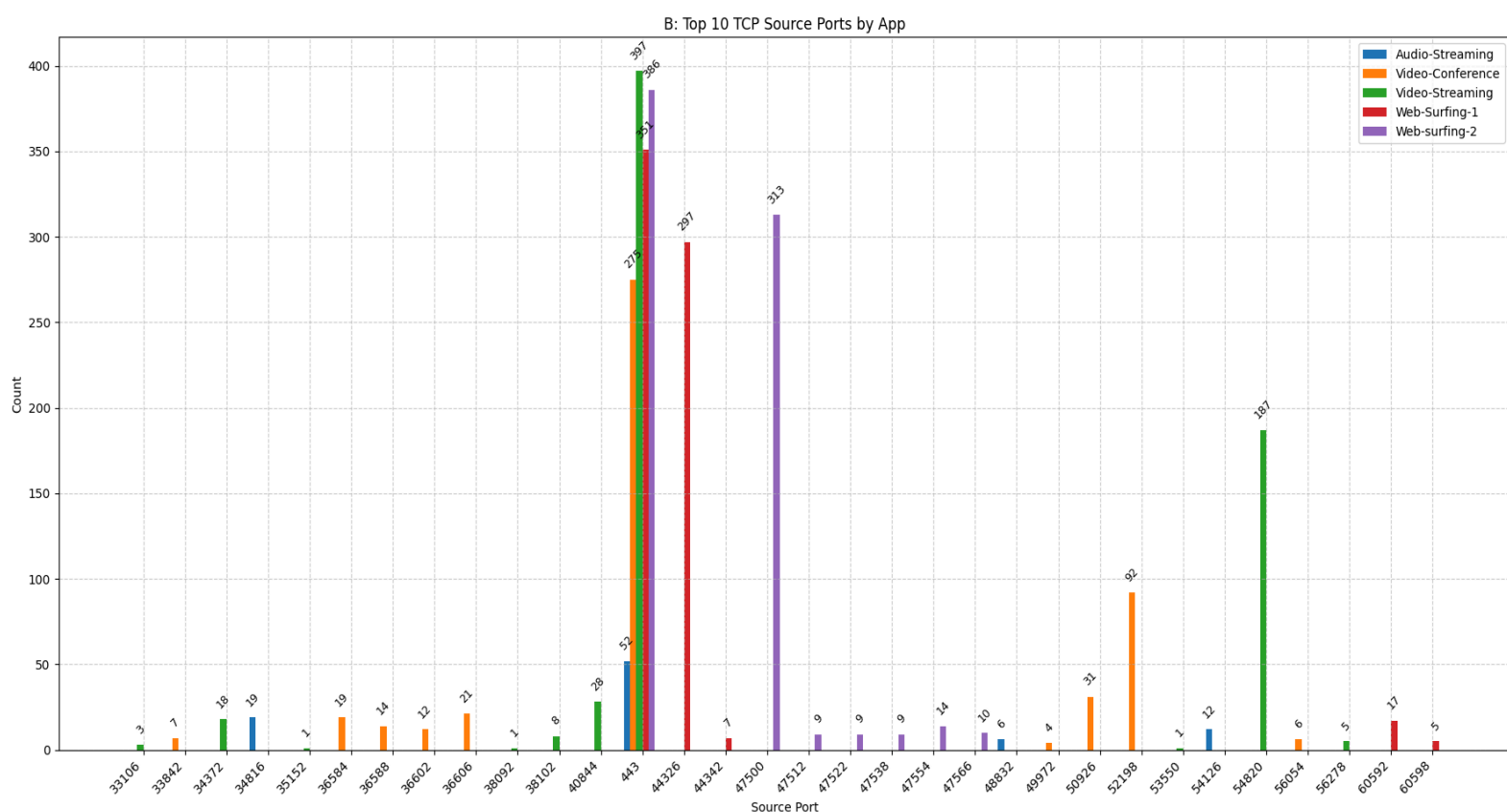
**80:** פורט ברירת מחדל עבור תעבורת HTTP לא מאובטחת.

**44326, 41940, 52198, 50026, 53684, 57760, 36584:** פורטים זמניים (Ephemeral Ports) – נוצרים באופן דינמי עבור תקשורת זמנית.

### ציר ה-Y (מספר החבילות):

מציין את מספר הפקטות (packets) שנשלחו דרך כל פורט עבור כל אפליקציה.

ספירה זו מאפשרת לראות אילו פורטים היו בשימוש הכי תדיר בתקשורת בין הלקוח לשרת.



## פירוט גרף C: התפלגות זמני הגעה בין-פקטות:

### מה מייצג הגרף?

הגרף מציג את הזמנים בין הגעת שתי פקטות עוקבות (Inter-arrival time) כנגד הצפיפות של הגעתן בחמשת האפליקציות שנבדקו:

.Web-surfing 1, Web-surfing 2, Audio Streaming, Video Streaming, Video Conference

זמן הגעת הפקטות משקף את דפוסי השימוש של כל אפליקציה ואת סוג התעבורה שלה.

### ציר ה-X (זמן הגעה בין-פקטות בשניות)

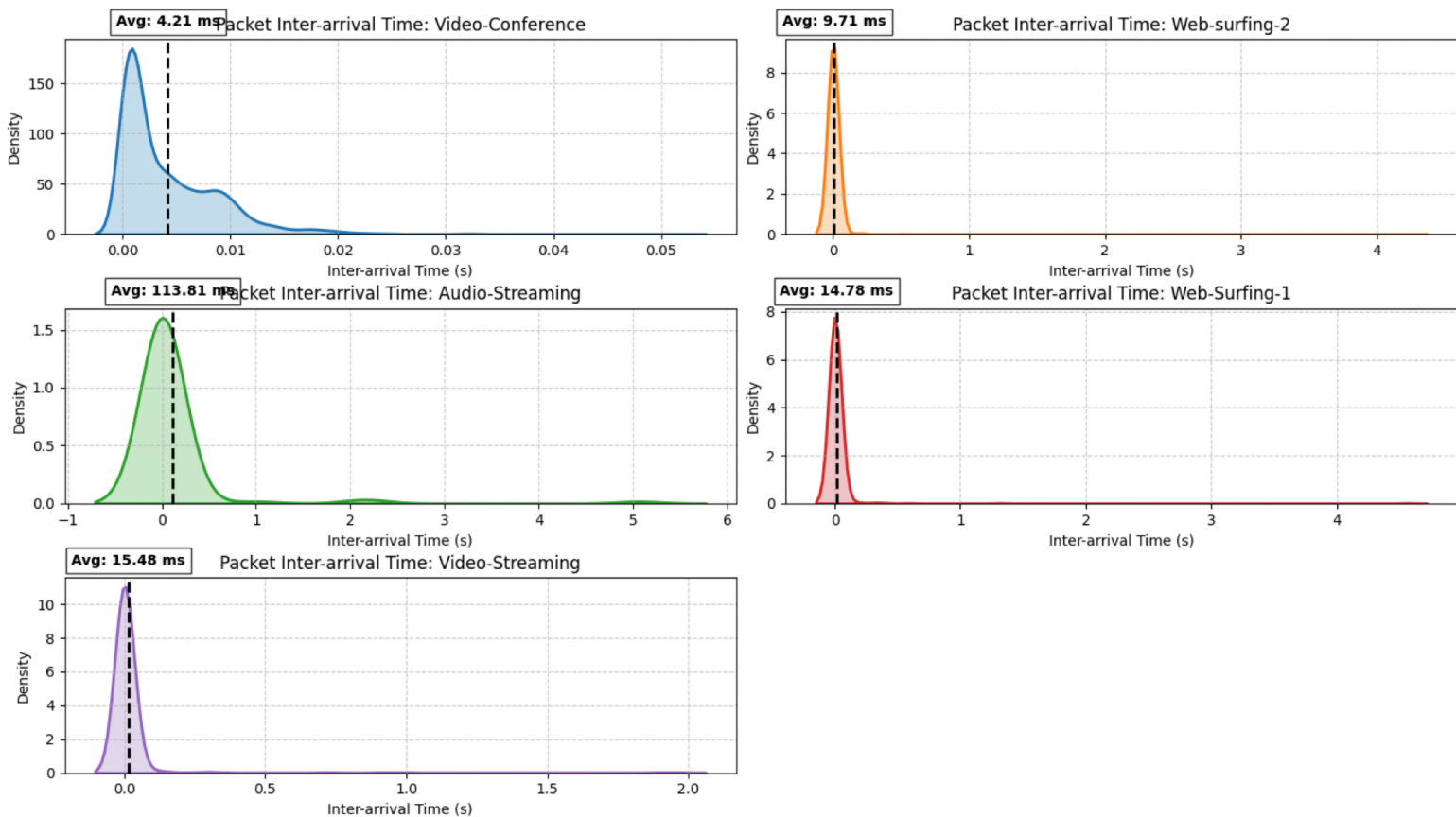
ציר ה-X מציין את הזמן בין כל שתי פקטות רצופות שנשלחו מהרשת ללקוח. הנתונים מוצגים בסקלה לוגריתמית (Log Scale), כיוון שהפערים יכולים לנוע בין מיקרו-שניות ( $\mu s$ ) לשניות (s).

ערכים קטנים יותר  $\leftarrow$  פקטות מגיעות בקצב מהיר יותר.  
ערכים גדולים יותר  $\leftarrow$  פקטות מגיעות בפערי זמן גדולים יותר (למשל, כאשר יש המתנה להורדת תוכן נוסף).

### ציר ה-Y (צפיפות):

מייצג את רמת השכיחות של זמנים מסוימים בין חבילות. קווי הצפיפות הגבוהים מצביעים על פרקי זמן בין חבילות שבהם התקשורת מרוכזת.

C: Packet Inter-arrival Time Distribution by App



## פירוט גרף D: התפלגות גודל החבילות:

### מה הגרף מציג?

הגרף מתאר את התפלגות גודל החבילות (Packet Size Distribution) כנגד תדירות הגעתן של החבילות בחמש אפליקציות שונות:

Video-Conference , Video-Streaming , Web-Surfing-1 , Web-Surfing-2, Audio-Streaming

המטרה של הגרף היא להראות כיצד כל אפליקציה משתמשת בגודל חבילות שונה, בהתאם לצורכי התעבורה שלה (רציפות, מהירות, אבטחה ואיכות שירות).

### ציר ה-X: גודל החבילה (Packet Size - Bytes):

מתאר את הגודל של כל חבילת מידע שנשלחה בתעבורה של האפליקציה.

החבילות נמדדות ב-Bytes (בתים) וכוללות:

כותרות (Headers) – מידע פרוטוקולי כמו IP, TCP, TLS.

נתוני תוכן (Payload) – המידע עצמו (שמע, וידאו, תוכן אינטרנט וכו').

קיים טווח רחב של גדלים – החל מחבילות קטנות מאוד (כמה עשרות Bytes) ועד חבילות גדולות מאוד (אלפי Bytes):

חבילות קטנות (0–500 Bytes): נפוצות בשיחות בזמן אמת, בהן המהירות קריטית.

חבילות בינוניות (500–2000 Bytes): אופייניות לשירותים שדורשים שילוב של מהירות ואמינות.

חבילות גדולות (2000–10,000 Bytes ומעלה): משמשות להורדת תוכן כבד כמו סרטונים או קבצים גדולים.

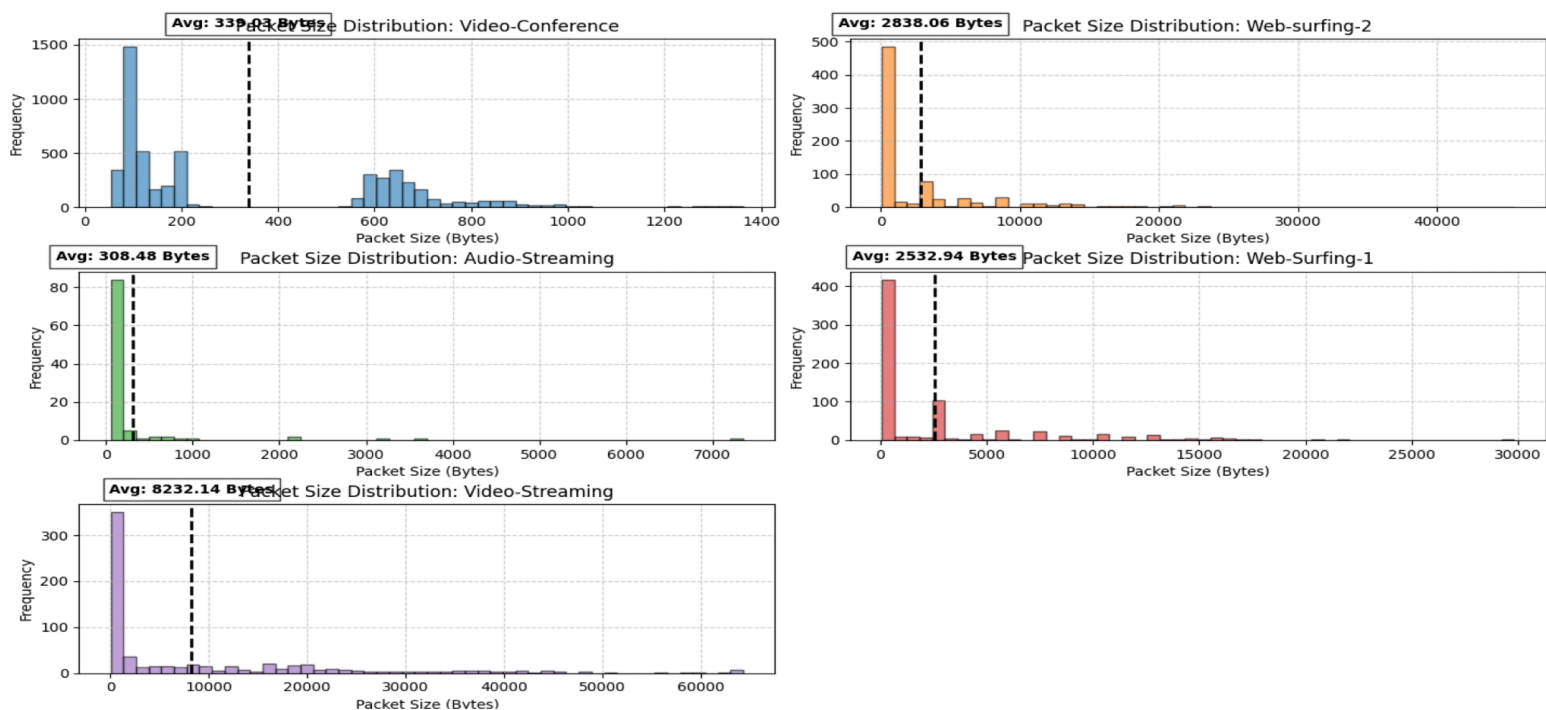
### ציר ה-Y: תדירות הופעת החבילה (Frequency):

מייצג את מספר הפעמים שבהם חבילה בגודל מסוים הופיעה במהלך ההקלטה.

ככל שהעמודה גבוהה יותר – כך החבילה בגודל זה הייתה נפוצה יותר.

תדירות גבוהה של חבילות קטנות מעידה על צורך בעיכוב מינימלי (Latency), בעוד תדירות גבוהה של חבילות גדולות מצביעה על אופטימיזציה של רוחב פס להעברת כמויות מידע גדולות בפרק זמן קצר.

D: Packet Size Distribution by App



## תיאור הגרפים והמסקנות:

### הסבר גרף A:

1. שיחות וידאו (Video Conference) מתבססות בעיקר על UDP ובנוסף DTLS מופיע באופן משמעותי  
Zoom משתמש בפרוטוקול UDP לניהול השיחה, אך בנוסף ניתן לראות שימוש משמעותי בפרוטוקול DTLS, מה מעיד על הצפנה חזקה ברמת התעבורה. הופעת DTLS מעידה על מנגנון אבטחה מוגבר, שבו הנתונים המועברים בזמן אמת מוצפנים, מה שמוסיף שכבת הגנה על המידע המשודר.
2. גלישה באינטרנט מתבצעת באמצעות TLS ו-HTTP/2  
Chrome ו-Firefox משתמשים בפרוטוקולים TLS ו-HTTP/2, המעידים על הצפנה מתקדמת ושימוש בטכנולוגיות מודרניות להעברת נתונים מאובטחת ויעילה יותר. Firefox מציג תעבורה גבוהה יותר בפרוטוקול TLS בהשוואה ל-Chrome, מה שעשוי להעיד על מנגנוני הצפנה שונים בין הדפדפנים. HTTP/2 מופיע בתעבורה של הדפדפנים, דבר שמצביע על אופטימיזציה בביצועי טעינת האתרים ושיפור מהירות ההעברה.
3. וידאו סטרימינג (YouTube) מתבצע באמצעות TLS ו-HTTP/2  
TLS משמש להצפנת התקשורת, בעוד HTTP/2 אחראי לניהול הזרמת הוידאו, דבר המעיד על שיפור ביצועים יעול תעבורת הנתונים. UDP לא מופיע בתעבורת YouTube, מה שמצביע על כך שההזרמה מבוססת על TCP + TLS ללא שימוש ב-QUIC. (ביטלנו אותו לצורך הניסוי)
4. אודיו סטרימינג (Spotify) מתבצע על גבי TLS וללא שימוש ב-UDP  
ספוטיפיי משתמש ב-TCP + TLS, מה שמעיד על הצפנה מלאה של תעבורת השמע. העדר UDP מעיד על כך שהזרמת האודיו מבוצעת בתקשורת מבוססת חיבור (TCP), ככל הנראה בשל מנגנוני שליטה טובים יותר באיכות ההשמעה. (וגם כי ביטלנו את QUIC).
5. שיחות וידאו (Zoom) עושות שימוש בפרוטוקול DTLS, המעיד על הצפנה חזקה יותר בתקשורת השמע והוידאו  
DTLS נוכח בתעבורה, דבר המצביע על שימוש במנגנון הצפנה חזק המגן על תקשורת בזמן אמת. בנוסף, Zoom עושה שימוש משמעותי ב-UDP, מה שמאפשר לו לספק תקשורת וידאו ושמע עם השהייה מינימלית.
6. STUN נמצא בשימוש במנגנוני NAT traversal  
STUN מופיע בתעבורה, ככל הנראה לצורך מעבר דרך נתבים וחומות אש. הופעת STUN בהקשר של Zoom ושירותים נוספים מראה שהפרוטוקול מסייע בהקמת חיבורים ישירים בין משתמשים.
7. כמות קטנה של תעבורת DNS, מכיוון שלא הייתה חשיבות לבקשות DNS רבות בתעבורה הנמדדת  
ניתן לראות כי ישנן מעט מאוד בקשות DNS, דבר שמצביע על כך שרוב התקשורת בתעבורה שנבדקה לא דרשה ריבוי שאילתות DNS.

מרבית התקשורת היא הזרמת נתונים (Streaming), גלישה ושיחות וידאו, ולכן אין צורך בבקשות **DNS** רבות, מאחר והחיבורים מתבצעים ישירות לשרתים הרלוונטיים לאחר שהדפדפן או היישום פתר את כתובת ה-IP בתחילת התהליך.

## הסבר גרף B:

### 1. שיחות וידאו (Video Conference) משתמשות בפורטים גבוהים (49152 ומעלה) ובפורטים ייעודיים אחרים

Zoom משתמש בעיקר בפורטים 52198 ו-50926, שהם פורטים זמניים. זה מראה שהאפליקציה משתמשת בפורטים דינמיים לחיבורי TCP, חלק ממנגנון ה-NAT Traversal. כמו כן, ניתן לראות שימוש נמוך בפורטים 36584, 36588 ו-36606, מה שעשוי להיות קשור לממשק הרשת של Zoom. (לאחר בדיקה בגוגל וחיפוש)

### 2. הזרמת וידאו (YouTube) משתמשת בפורטים הקלאסיים של HTTP ו-HTTPS

ניתן לראות שימוש משמעותי בפורטים 443 ו-40844, מה שמצביע על שימוש בפרוטוקול HTTPS. פורט 443 הוא הסטנדרטי עבור TLS, ולכן השימוש הנרחב בו הגיוני. בנוסף, ניתן לראות שימוש בפורט 54820, שהוא פורט זמני שמוקצה על ידי מערכת ההפעלה עבור חיבורי TCP. (גם לאחר חיפוש על הפורט)

### 3. גלישה באינטרנט (Web-Surfing) מציגה שימוש דומה בין Firefox ו-Chrome, עם הבדל קל בפורטים המשניים

גם Chrome וגם Firefox משתמשים באופן משמעותי בפורט 443, שכן רוב אתרי האינטרנט כיום משתמשים ב-HTTPS. עם זאת, Firefox נוטה להשתמש גם בפורט 47500, בעוד ש-Chrome משתמשת יותר בפורט 44326. ההבדל הזה נובע ממדיניות ההקצאה של פורטים זמניים בדפדפנים או מהשרתים אליהם התחברו.

### 4. הזרמת אודיו (Spotify) משתמשת בעיקר בפורטים של HTTPS ופורטים זמניים

ניתן לראות שימוש בפורטים 443, 54126 ו-34816. פורט 443 מעיד על שימוש בפרוטוקול TLS עבור אבטחת הזרמת הנתונים. בנוסף, שימוש בפורטים זמניים מראה לנו ש-Spotify מקצה חיבורים דינמיים להזרמה ולא משתמשת בפורטים קבועים.

### 5. נראה שימוש ב-STUN באמצעות הפורטים 3478, 5349 או 44326

פורטים אלו משמשים לרוב לפרוטוקול STUN, שמטרתו היא לאפשר חיבורי תקשורת בין לקוחות מאחורי NAT. השימוש בפורטים זמניים נוספים כגון 52198, 50926, 56054 עשוי להעיד על כך שהאפליקציות מבצעות חיבורי P2P מסוימים.

## הסבר גרף C:

### 1. שיחות וידאו (Video Conference) מציגות זמני הגעה קצרים מאוד של חבילות, מה שמעיד על שימוש בפרוטוקול מבוסס UDP.

ניתן לראות שבשיחות וידאו (Zoom) זמני ההגעה בין החבילות הם הקצרים ביותר (כ-4.21 מילישניות בממוצע).



הדבר מצביע על כך שהחבילות נשלחות באופן כמעט מיידי זו אחר זו, מה שמאפיין תקשורת אינטראקטיבית הדורשת השהיות נמוכות. השימוש ב-UDP מאפשר העברת חבילות מהירה ללא הצורך בהמתנה לאישור קבלה, ולכן הפרוטוקול מתאים במיוחד לשיחות וידאו.

## 2. הזרמת וידאו (YouTube) מציגה זמני הגעה ממוצעים נמוכים יחסית, אך ארוכים יותר מאשר בשיחות וידאו.

זמני ההגעה של חבילות בהזרמת וידאו (כ-15.48 מילישניות בממוצע) גבוהים מאלו של שיחות וידאו אך עדיין נמוכים יחסית. הדבר מעיד על כך שהוידאו מוזרם בחבילות בצורה שוטפת אך עם מעט יותר השהיות בין החבילות, מה שעשוי להעיד על מנגנון אגירה (buffering) בתקשורת. מכיוון שהתעבורה אינה חייבת להגיע בזמן אמת כמו בשיחות וידאו, יתכן שימוש ב-TCP לצד UDP כדי להבטיח איכות שידור טובה יותר.

## 3. גלישה באינטרנט (Web-Surfing) מציגה שוני כלשהו בין הדפדפנים, כאשר Firefox מציג זמני הגעה קצרים יותר מ-Chrome.

בזמני ההגעה הממוצעים בין החבילות של פיירפוקס (ms 9.71) ניתן לראות כי הדפדפן משדר בקשות וחבילות בצורה יעילה יותר יחסית. לעומת זאת, Chrome מציג זמני הגעה ממוצעים מעט גבוהים יותר (ms 14.78), מה שעשוי להעיד על מנגנוני ניהול חיבור שונים או שיטת אופטימיזציה שונה בשכבות הפרוטוקולים של הדפדפן. ניתן להניח שההבדלים בין הדפדפנים נובעים מהתנהגות ניהול הקשרים ומהשימוש בפרוטוקולים שונים. (או גם תוספים שונים שקיימים בדפדפנים השונים).

## 4. הזרמת אודיו (Spotify) מציגה זמני הגעה גבוהים באופן משמעותי בהשוואה לכל שאר היישומים, מה שמעיד על אופי תעבורת TCP.

זמני ההגעה הממוצעים בין החבילות ב-Spotify עומדים על 113.81 מילישניות, גבוהים בהרבה מכל שאר האפליקציות. הדבר מצביע על כך שהתעבורה מתבססת בעיקר על TCP, שבו יש צורך באישורים חוזרים על כל חבילה שנשלחת. התנהגות זו הגיונית בהזרמת אודיו, מכיוון שהיישום משתמש באגירה של חבילות (buffering) ואינו דורש תעבורה מיידיה כמו בשיחות וידאו. בהשוואה להזרמת וידאו (YouTube), ניתן לראות שהפרשי זמני ההגעה מצביעים על כך שספוטיפיי פחות רגיש לאיבוד חבילות ולכן לא מחויב להשתמש ב-UDP.

## 5. באופן כללי, ניתן לראות כי אפליקציות מבוססות UDP מציגות זמני הגעה נמוכים יותר, בעוד אפליקציות מבוססות TCP מציגות זמני הגעה גבוהים יותר.

השוואה בין Zoom (שימוש נרחב ב-UDP) לעומת Spotify (שימוש ב-TCP) מדגימה את ההבדל המרכזי בין פרוטוקולי תקשורת בזמן אמת לעומת תקשורת מבוססת אמינות. ניתן להסיק כי יישומים הדורשים אינטראקטיביות גבוהה יעדיפו להשתמש בפרוטוקול UDP על פני TCP כדי למנוע השהיות.

### D.3.3 הסבר גרף D:

## 1. שיחות וידאו (Video Conference) מציגות חבילות קטנות באופן יחסי, מה שמעיד על שימוש בפרוטוקול UDP.

ניתן לראות כי החבילות הנפוצות ביותר בשיחות וידאו (Zoom) הן בגודל ממוצע של 339.03 בתים.

השימוש בחבילות קטנות נובע מכך ששיחות וידאו מחייבות תעבורה רציפה ומהירה, ולכן עדיף לשלוח חבילות קטנות ובתדירות גבוהה כדי להימנע מהשהיות. הדבר תואם את העובדה שהשירות משתמש בפרוטוקול UDP, שמאפשר שליחה רציפה של מידע ללא צורך בהמתנה לאישור קבלה.

## 2. בהזרמת וידאו (YouTube) נקבל חבילות גדולות יותר, מה שמעיד על השימוש ב-TCP ובפרוטוקולי הצפנה כגון TLS.

החבילות הנפוצות ביותר בהזרמת וידאו הן בגודל ממוצע של **8232.14 בתים**, מה שמרמז על שליחה אופטימלית של נתוני וידאו דחוסים בפרוטוקול אמין. הזרמת וידאו אינה דורשת זמן אמת כמו שיחות וידאו, ולכן ניתן לשלוח חבילות גדולות יותר כדי לנצל בצורה מיטבית את רוחב הפס. עובדה זו מעידה על שימוש ב-TCP עם TLS להבטחת אמינות ההעברה והצפנת הנתונים.

## 3. גלישה באינטרנט (Web-Surfing) מציגה הבדלים בין דפדפנים, כאשר Firefox משתמש בחבילות מעט גדולות יותר מ-Chrome.

בגישה לאתר מאקו, החבילות הממוצעות ב-Firefox הן בגודל **2838.06 בתים**, בעוד שב-Chrome החבילות הממוצעות קטנות יותר ועומדות על **2532.94 בתים**. מה שיכול לנבוע מהבדלים בניהול החיבורים של הדפדפנים / הבדלים בכיוון תעבורה / מנגנוני טעינה מוקדמת.. ניתן לראות כי בשני הדפדפנים, גודל החבילות הממוצע קטן מהותית מזה של הזרמת וידאו, מה שמעיד על כך שתעבורת HTTP/S מורכבת מבקשות קצרות ותשובות מותאמות.

## 4. הזרמת אודיו (Spotify) מציגה חבילות קטנות יחסית, אך גדולות יותר מאלו של שיחות וידאו.

החבילות הנפוצות ביותר בהזרמת אודיו עומדות על ממוצע של **308.48 בתים** לחבילה, גבוהות מעט מאלו של שיחות וידאו, אך קטנות משמעותית מאלו של הזרמת וידאו (Youtube). הדבר נובע מכך ששירותי הזרמת אודיו משתמשים ב-TCP עם מנגנוני אגירה (buffering), המאפשרים טעינה מוקדמת של תוכן כדי למנוע השהיות בשידור. בניגוד לשיחות וידאו, ההזרמה אינה דורשת עדכונים מיידיים ולכן ניתן להעביר את המידע בחבילות מעט גדולות יותר.

## 5. באופן כללי, ניתן לראות כי אפליקציות מבוססות UDP משתמשות בחבילות קטנות יותר, בעוד אפליקציות מבוססות TCP משתמשות בחבילות גדולות יותר.

ההשוואה בין Zoom (שימוש ב-UDP) לבין YouTube ו-Spotify (שימוש ב-TCP) ממחישה את השוני בין פרוטוקולי התקשורת. שיחות וידאו דורשות שליחת חבילות קטנות בתדירות גבוהה, בעוד שהזרמות מדיה (אודיו/וידאו) מסתמכות על חבילות גדולות יותר כדי לנצל את רוחב הפס בצורה מיטבית. עובדה זו מחזקת את ההנחה כי יישומים אינטראקטיביים יעדיפו UDP, בעוד שיישומים המבוססים על הזרמה אמינה יעדיפו TCP.

### 3.4. סימולציה של תקיפה לזיהוי אפליקציות על בסיס תעבורה מוצפנת

הקדמה והסבר:

בסעיף הזה מבקשים מאיתנו לשחק תפקיד של תוקף שמנסה לזהות באילו אפליקציות או אתרים המשתמש ביקר, גם אם התעבורה מוצפנת. יש שתי אפשרויות שאליהן נחשף התוקף:

1. **אפשרות א': התוקף יודע את גודל הפקטות, חותמת הזמן שלהן, ואת ה-hash של ה-tuple-4 (כתובת IP מקור, כתובת IP יעד, פורט מקור, פורט יעד):**

במקרה זה, התוקף יכול לקבץ את התעבורה לזרמים (Flows) לפי שילוב כתובת ה-IP והפורט.

אם התוקף מזהה תבנית גודל חבילות מסוימת או דפוס של תדירות זמנים קבועה שניתן לאפיין בין חבילות – ניתן להסיק איזו אפליקציה או שירות נמצאים בשימוש.

לדוגמה:

**Zoom** ישלח הרבה פקטות קטנות בתדירות גבוהה (שיחות וידאו).

**YouTube** ישלח פקטות גדולות בהפרשי זמן קבועים (באפרינג של וידאו).

**Spotify** ישלח פקטות בגודל בינוני עם הפרשי זמן שונים. (גם מבצע באפרינג אך פחות משמעותי)

2. **אפשרות ב': התוקף יודע רק את גודל הפקטות ואת חותמות הזמן שלהן:**

כאן, התוקף לא יכול לקבץ זרמים בקלות כי אין לו מידע על ה-IP והפורט.

עם זאת, ניתן לנסות לבצע ניתוח סטטיסטי על דפוסי הגודל והתזמון ולזהות אם הם מתאימים לאפליקציה מסוימת.

לדוגמה, תוקף יכול לגלות שימוש ב-YouTube אם הוא רואה חבילות גדולות כל כמה שניות – גם בלי לדעת לאן הן נשלחות.

לעומת זאת, שימוש ב-Zoom יופיע כחבילות קטנות עם תדירות גבוהה, גם אם התוקף לא יודע מה היעד.

## נסביר איך כל קריטריון יתרום לתוקף בעת נסיונו לזהות את התעבורה:

### 1. חותמת זמן (Timestamp)

חותמת זמן היא הרגע שבו הפקטה נשלחה או התקבלה, והיא משמשת כדי להבין את סדר הפקטות ואת המרווחים ביניהן.  
לדוגמה:

Timestamp: 1711782923.451

ניתן לחשב מרווחים בין פקטות עוקבות כדי להבין תדירות התקשורת של אפליקציה מסוימת.

### איך חותמת זמן עוזרת לתוקף?

אם פקטות נשלחות במרווחים קבועים (כמו 0.3 שניות), זה עשוי להעיד על הזרמת וידאו (buffering).  
אם יש פרצי חבילות (bursts) בזמן קצר מאוד, ייתכן שמדובר בשיחת וידאו (Zoom).  
חבילות בודדות עם הפרשים גדולים יכולות להיות טעינת אתר בדפדפן (כמו שכבר ראינו)

### 2. Hash של ה-Tuple-4

ה-Tuple-4 (רביעייה) מתאר מאפיינים ייחודיים לכל זרם תקשורת (Flow) והוא מורכב מ:  
Source IP, Destination IP, Source Port, Destination Port.

לדוגמה, אם משתמש פותח חיבור HTTPS ל-Youtube, ייתכן שנראה משהו כזה:

(Source IP: 192.168.1.100, Destination IP: 142.250.187.46, Source Port: 52000, Destination Port: 443)

כדי לטשטש את הכתובות, ניתן להמיר את הרביעייה ל-hash כך שהתוקף לא יידע את הכתובות בפועל, אך עדיין יוכל לזהות פקטות ששייכות לאותו חיבור.

לדוגמה:

$\text{hash}(192.168.1.100, 142.250.187.46, 52000, 443) \rightarrow "A7F9D8C3B2"$

כל הפקטות שמשתמשות באותו חיבור יקבלו את אותו ה-hash, וכך התוקף יכול לזהות זרמים (Flows) מבלי לדעת בפועל את כתובות ה-IP.

### איך ה-Hash של ה-Tuple-4 עוזר לתוקף?

הוא מזהה חיבורים שונים גם אם ה-IP מוצפן או מוסתר.  
הוא מבדיל בין תקשורת מקבילה, למשל:  
אחד לשיחת Zoom  
אחד להזרמת מוזיקה בספוטיפיי  
אחד לגלישה ב-Chrome  
הוא מאפשר ניתוח של זרמים שונים (Flows) ולראות אילו פקטות שייכות לאותה פעילות.

### 3. Flow ID:

נתחיל בלהסביר מה הכוונה ב Flow:

#### **Flow (זרם תקשורת):**

כל חיבור נפרד בין שני מכשירים נחשב Flow.

**לדוגמה**, אם אתה צופה ב-Youtube בזמן שאתה משתמש ב-Zoom, אז יש לפחות שני Flows שונים:

Zoom: תקשורת אינטנסיבית עם הרבה חבילות UDP קטנות.

Youtube: הורדת חבילות גדולות מהשרת של Youtube, את זה ניתן לעשות בגלל השימוש ב Buffering.

### Flow ID:

ה-Flow ID הוא מזהה ייחודי לכל זרם תקשורת והוא לרוב מחושב על בסיס ה-Tuple-4:

$$\text{Flow ID} = \text{hash}(\text{Source IP}, \text{Destination IP}, \text{Source Port}, \text{Destination Port})$$

כך שאם יש לדוגמא חיבור למשתמש עם Youtube:

$192.168.1.10 \rightarrow 142.250.187.46:443$

ה-Flow ID עבורו יהיה שונה מה-Flow ID של שיחת Zoom שמתרחשת באותו זמן.

#### **איך Flows עוזרים לתוקף?**

אם התוקף מזהה מספר רב של Flows קצרים מאוד (Zoom משתמש בהרבה חיבורים קצרי טווח), הוא יכול לשער שמדובר בשיחת וידאו.

אם יש Flow אחד עיקרי עם חבילות גדולות, ייתכן שמדובר בהזרמת וידאו (YouTube, Netflix).

אם Flows נמשכים זמן רב אך עם תעבורה מעטה, מדובר כנראה בגלישה באינטרנט (Web-Surfing).

כל זה כמובן מבוסס על מה שהראנו בשאלה 3 בגרפים.

(\*על גודל הפקטות לא צריך להסביר לכן נסביר רק איך הוא תורם לתוקף)

### 4. גודל הפקטות

#### **איך גודל הפקטות עוזר לתוקף?**

הגודל של הפקטות מספק מידע חשוב, גם אם התוכן מוצפן. מהמסקנות שהסקנו בחלק הקודם נובע:

#### **1. פרוטוקולים שונים שולחים פקטות בגדלים אופייניים**

פקטות VoIP (שיחות קוליות) נוטות להיות קטנות (100-200 בתים).

פקטות הזרמת וידאו נוטות להיות גדולות (1000-1500 בתים) כדי לשדר מידע רב.

תעבורת **Web** מכילה פקטות קטנות (בקשות HTTP) ופקטות גדולות (תשובות עם תוכן האתר).

## 2. תבניות תעבורה לפי סוג היישום

גלישה באתרי חדשות תראה גודל פקטות מגוון מאוד.

יוטיוב או נטפליקס יראו גודל פקטות גדול באופן אחיד.

שירותי הודעות מידיית (WhatsApp, Telegram) ישלחו פקטות קטנות עם מרווחי זמן גדולים ביניהן.

## 3. ניתוח יחס בקשות-תשובות

בבקשות HTTP/S ישנה בקשה קטנה מאוד (Client Request) ותשובה גדולה יחסית (Server Response).

בפרוטוקולים כמו VoIP, גודל הפקטות נותר קבוע וקטן.

## מדוע זה שימושי לתוקף?

תוקף יכול להשתמש באותו ניתוח כדי לזהות **חתימה סטטיסטית של כל אפליקציה**.

גם אם התעבורה **מוצפנת**, גודל הפקטות ותזמון מספקים מידע חשוב לזיהוי השירותים בהם המשתמש משתמש.

ניתן לשפר את ההתקפה עם Machine Learning מתקדם כדי לזהות אפליקציות בדיוק גבוה.

## **פיתוח אלגוריתם לפי תכונות באפשרות א' - סיווג אפליקציות בהתבסס על גודל חבילות, חותמות זמן וזיהוי זרמים:**

דרישות וחבילות שעל הבודק להתקין בשביל שיוכל להשתמש בקוד בצורה תקינה :

Pandas

Os

Numpy

Matplotlib

Seaborn

Sklearn (scikit-learn)

Ace-tools

## תיאור האלגוריתם:

### 1. טעינת הנתונים מקבצי ה-CSV

הקוד טוען את נתוני התעבורה מקבצי ה-CSV שנמצאים בתיקייה **csv-files**.  
לכל קובץ, אנו שולפים:

- חותמות זמן (Timestamps) – לתיעוד מתי נשלחה כל חבילה.
- גודל הפקטות (Packet Size) – כדי להבין את נפח התעבורה.
- מזהה זרם (Flow ID) – לזיהוי הזרם אליו שייכת כל חבילה.

הנתונים עוברים ניקוי ראשוני, הכולל:

- המרת נתוני זמן לערכים מספריים.
- הסרת נתונים חסרים כדי להבטיח תקינות.

### 2. סיווג תעבורה לפי דפוסים

בשלב זה נזהה תבניות בתעבורה, שיעזרו לנו לשייך זרמים לסוגי אפליקציות:

- מרווחים בין פקטות קבועים → שיחת VoIP.  
כאשר מרווחי הזמן בין חבילות עקביים ונמוכים, זה עשוי להעיד על תעבורה של שיחות וידאו (Video Calls).
- גודל פקטות אחיד וגדול → הזרמת וידאו (Streaming).
- סטרימינג מתאפיין בחבילות גדולות יחסית, עם שונות נמוכה בגודל החבילות.
- שילוב של בקשות קטנות ותשובות גדולות → גלישה באינטרנט (Web Browsing).
- גלישה בדפדפן כוללת בקשות קטנות (GET, POST) עם תגובות גדולות יותר שמכילות תוכן.

### 3. חישוב מאפיינים סטטיסטיים לכל זרם

לאחר עיבוד הנתונים, מחשבים סט מאפיינים לכל זרם תעבורה, הכולל:

#### 3.1 גודל חבילה ממוצע וסטיית תקן

מאפשר זיהוי אפליקציות עם תנועה קבועה (כגון שיחות קוליות, שבהן הפקטות בגודל אחיד). גלישה באינטרנט מציגה שונות גבוהה בגודל החבילות עקב שילוב של בקשות קטנות ותשובות גדולות.

#### 3.2 מרווח זמן ממוצע וסטיית תקן בין חבילות

שיחות VoIP וסטרימינג אודיו יציגו מרווחים קבועים וצפופים. גלישה באינטרנט ויישומים אחרים יציגו שונות גדולה יותר בזמני ההגעה בין חבילות.

#### 3.3 מדד "התפרצות" התעבורה (Burstiness Measure)

מוגדר כיחס בין סטיית התקן של זמני ההגעה לבין הממוצע שלהם. תעבורה יציבה כמו סטרימינג תציג ערך נמוך, ואילו גלישה או תעבורה אקראית תציג ערכים גבוהים יותר.

#### 3.4 גודל חבילה מקסימלי

סטרימינג של וידאו ואודיו נוטה לכלול חבילות גדולות מאוד. גלישה ושירותי תקשורת (כגון צ'אטים) מכילים חבילות קטנות יותר.

#### 3.5 מספר הפקטות הכולל בזמן קבוע

זרמים אינטנסיביים כמו סטרימינג מכילים הרבה חבילות בפרק זמן נתון. גלישה באינטרנט כוללת פחות חבילות באופן יחסי.



#### **4. שימוש ב-K-Means Clustering לזיהוי אפליקציות**

לאחר חישוב כל המאפיינים, אנו מפעילים אלגוריתם K-Means Clustering על הנתונים. המטרה: לבדוק אם ניתן לשייך זרמי תעבורה לאפליקציות שונות על סמך דמיון סטטיסטי.

1. מחלקים את הנתונים לקבוצות - Clusters
2. מציגים את התוצאות על ציר דו-ממדי, מקודדות בצבעים שונים

## תוצאות:

כאשר ננסה לבצע את הסיווג על חמשת החבילות שהקלטנו בסעיף א' כאשר הן מכילות את כל המידע, נפתח אותן מהתיקיה csv\_files.

לאחר מכן באמצעות האלגוריתם שיצרנו ולאחר שאימנו את המודל שיצרנו נקבל את התוצאות הבאות :

```
=== CLASSIFICATION REPORT ===
```

	precision	recall	f1-score	support
Audio Streaming	0.90	1.00	0.95	9
Video Calls	1.00	1.00	1.00	12
Video Streaming	1.00	0.50	0.67	2
Web Browsing	0.00	0.00	0.00	1
accuracy			0.92	24

Accuracy: 0.92

בדוח הסיווג מוצגות תוצאות עבור ארבע קטגוריות עיקריות (Audio Streaming, Video Calls, Video Streaming, Web Browsing). לכל קטגוריה מופיעים המדדים:

- Precision (דיוק): מתוך כלל הדגימות שסווגו לקטגוריה זו, כמה באמת שייכות אליה בפועל.
- Recall (רגישות): מתוך כלל הדגימות שבאמת שייכות לקטגוריה, כמה זוהו נכונה.
- Support: מספר הדגימות בפועל בכל קטגוריה.
- F1 : הוא ממוצע הרמוני שמאזן בין דיוק (Precision) לבין זיהוי נכון (Recall), ומספק מדד יחיד המשקף את ביצועי הסיווג.

בסוף הדוח מצוין ה-Accuracy (דיוק כללי) של 0.92, כלומר 92% מהדגימות סווגו נכונה בכלל הקטגוריות. ערכי Precision ו-Recall גבוהים מעידים על מודל שיודע גם למנוע סיווג-יתר (Precision גבוה) וגם לזהות את מרבית הדגימות הרלוונטיות (Recall גבוה).

**כלומר , הצלחנו לסווג את כל המידע לפי סוג התעבורה שלו כאשר אנחנו יודעים את כל הקריטריונים הנדרשים וכאשר אין הפרעות תעבורה ברקע כמו שהסעיף ביקש מאיתנו.**

## סקירה כללית

בתרחיש זה אנו מדמים מתקפה שבה לתוקף יש גישה מוגבלת מאוד לנתוני התעבורה. התוקף יודע רק שני פרמטרים על כל חבילה:

1. גודל החבילה (Packet Size)
2. חותמת זמן (Timestamp)

המטרה היא לבדוק עד כמה תוקף יכול לזהות אילו אפליקציות או אתרים המשתמש מבקר, גם כאשר התעבורה מוצפנת או אנונימית.

## מגבלות התוקף

מכיוון שהתוקף יודע רק את גודל החבילה ואת חותמות הזמן, יש מידע רב שהוא אינו יכול לגשת אליו:

1. אין לו מידע על כתובות ה-IP של המקור והיעד.
2. אין לו מידע על מזהה זרם (Flow ID) המבוסס על ה-4 Tuple (source IP, dest IP, source port, dest port).
3. אין לו גישה לתוכן החבילות, שכן התעבורה מוצפנת.

עם זאת, התוקף יכול להשתמש בדפוסים מסוימים בגודל החבילות ובמרווחי הזמן ביניהן כדי לנחש את סוג האפליקציה.

## אופן הביצוע

בסעיף ב ניצור אלגוריתם דומה לסעיף א שמנתח את אותה התעבורה שניתחנו בסעיף א, אך, הוא משתמש הפעם בתעבורה הלא מוצפנת ומנתח אותה כאשר לא נעשה שימוש תעבורה המפוענחת, אלא רק בתעבורה המוצפנת, ובנוסף יש לו גישה רק לגודל החבילות ולחותמת הזמן של כל חבילה, ניצור בדיוק את אותם קבצי csv, ללא שימוש בחבילות המפוענחות וללא העמודות המתאימות (נצרף דוגמא להבדל).

להלן קובץ ה-csv המתאר את web-browsing-1 בסעיף א (המכיל את הפקטות לאחר פתיחת ההצפנה):

A	B	C	D	E	F	G
No.	Time	Source	Destination	Protocol	Length	Info
1	0	192.168.12	192.168.12	DNS	74	Standard query 0x6bcb A www.mako.co.il
2	0.000604	192.168.12	192.168.12	DNS	74	Standard query 0x85c1 HTTPS www.mako.co.il
3	0.005976	192.168.12	192.168.12	DNS	211	Standard query response 0x85c1 HTTPS www.mako.co.il CNAME wilcard.mako.co.il.edgekey.net CNAME e974.b.akamaiedge.net SOA n0b.akamaiedge.net
4	0.006312	192.168.12	192.168.12	DNS	166	Standard query response 0x6bcb A www.mako.co.il CNAME wilcard.mako.co.il.edgekey.net CNAME e974.b.akamaiedge.net A 23.56.225.242
5	0.012359	192.168.12	23.56.225	TCP	74	44326 > 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=3058958283 TSecr=0 WS=128
6	0.01295	192.168.12	23.56.225	TCP	74	44342 > 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=3058958283 TSecr=0 WS=128
7	0.059978	23.56.225	192.168.12	TCP	60	443 > 44342 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
8	0.059978	23.56.225	192.168.12	TCP	60	443 > 44326 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
9	0.060146	192.168.12	23.56.225	TCP	54	44342 > 443 [ACK] Seq=1 Ack=1 Win=64240 Len=0
10	0.060275	192.168.12	23.56.225	TCP	54	44326 > 443 [ACK] Seq=1 Ack=1 Win=64240 Len=0
11	0.06116	192.168.12	23.56.225	TLSv1.3	2081	Client Hello (SN=www.mako.co.il)
12	0.061508	23.56.225	192.168.12	TCP	60	443 > 44342 [ACK] Seq=1 Ack=1461 Win=64240 Len=0
13	0.061508	23.56.225	192.168.12	TCP	60	443 > 44342 [ACK] Seq=1 Ack=2028 Win=64240 Len=0
14	0.061817	192.168.12	23.56.225	TLSv1.3	2081	Client Hello (SN=www.mako.co.il)
15	0.06202	23.56.225	192.168.12	TCP	60	443 > 44326 [ACK] Seq=1 Ack=1461 Win=64240 Len=0

מנגד קובץ ה-csv שמתאר את web-browsing-1 בסעיף ב' (שמכיל תעבורה מוצפנת בלבד):

A	B	C
No.	Time	Length
1	0	74
2	0.000604	74
3	0.005976	211
4	0.006312	166
5	0.012359	74
6	0.01295	74
7	0.059978	60
8	0.059978	60
9	0.060146	54
10	0.060275	54
11	0.06116	2081
12	0.061508	60
13	0.061508	60
14	0.061817	2081
15	0.06202	60

## תוצאות:

```

=== CLASSIFICATION REPORT (LIMITED DATA) ===
Class                Precision    Recall    Support
-----
Audio Streaming      1.00      0.00      1
Video Calls          1.00      0.00      1
Video Streaming      1.00      0.00      1
Web Browsing         0.40      1.00      2

Accuracy: 0.40

```

בדוח הנוכחי ניתן לראות ארבע קטגוריות (Audio Streaming, Video Calls, Video Streaming, Web Browsing), כאשר מספר הדגימות (Support) בכל קטגוריה נמוך מאוד: קטגוריות מסוימות מופיעות פעם אחת בלבד, ו-Web Browsing פעמיים. כתוצאה מכך, דיוק המדדים (Precision, Recall) מושפע מאוד מכל טעות בודדת, ומתקבלות תוצאות כמו Recall=0 בקטגוריות עם דגימה בודדת שלא זוהתה נכון. כך, למרות שבחלק מהקטגוריות המדד Precision הוא 1.00 (כשדגימה אחת סווגה נכון), במקרים אחרים הוא 0.00 כאשר אותה דגימה הוחמצה. התוצאה הסופית (Accuracy=0.40) מעידה על כך שהמודל מצליח לנבא נכונה רק כ-40% מהדגימות הכוללות, ניתן להסיק שהיעדר מידע נוסף (כמו כתובות מקור/יעד או פרוטוקול) מקשה על המודל להבדיל בין סוגי התעבורה, במיוחד כאשר יש מעט דגימות בכל קטגוריה.

## מסקנה מהתוצאות:

במודל הקודם (שנעזר ב-Flow ID ובארבעה שדות: כתובות IP מקור/יעד ופורטים), כל CSV יכול היה להכיל מספר "זרמים" (Flows) שונים. בכל פעם שהקוד איתר הבדל ב-Tuple-4 (מקור, פורט מקור, יעד, פורט יעד) או בפרוטוקול, נוצר Flow חדש. כך, גם אם עבדנו על אותם קבצי CSV, הקוד "פיצל" את המידע למספר דגימות רב יותר — אחת לכל זרם.

לעומת זאת, בקוד שמבוסס רק על שני עמודות (Time, Length), אנו מתייחסים לכל CSV כאל "דגימה" יחידה, ללא חלוקה פנימית לזרמים נפרדים. כתוצאה מכך, מספר הדגימות הכולל קטן יותר (ולכן גם ערכי ה-Support בכל קטגוריה נמוכים יותר). כלומר, במודל הקודם "נוצרו" יותר דגימות מפני שכל CSV פוצל למספר Flows; במודל הנוכחי (רק שני שדות), אין מנגנון פיצול, ולכן כל CSV הוא דגימה אחת.

## מסקנות לגבי מידע שהתוקף יכול לפענח:

- 1.1. **גודל החבילה:** פרוטוקולים ושירותים שונים משתמשים בתבניות גודל חבילות אופייניות. לדוגמה:
  - 1.1.1. **שירותי סטרימינג כמו YouTube או Netflix** נוטים לשלוח חבילות גדולות באופן עקבי.
  - 1.1.2. **שיחות VoIP כמו Zoom או Skype** משתמשות בחבילות קטנות מאוד הנשלחות בקצב קבוע.
  - 1.2. **שירותי גלישה באינטרנט** יכולו שילוב של חבילות קטנות (בקשות HTTP/S) וחבילות גדולות יותר (תשובות מהשרת).
  - 1.3. **סטטיסטיקות מתקדמות** (Flow Analysis, Burstiness)
- מדדים מתקדמים כמו Flow Entropy מאפשרים לזהות אילו שירותים פועלים במערכת שלך.  
**דוגמאות:**
  1. **גלישה באינטרנט** → דפוס לא סדיר עם חבילות גדולות וקטנות.
  2. **משחקים מקוונים** → זרימה קבועה עם מעט מאוד נתונים.
  3. **העברת קבצים (BitTorrent, Google Drive)** → חבילות גדולות מאוד שנשלחות בקצב יציב.
- 1.4. **חותמות זמן (Timestamps):** ניתוח זמני ההגעה של החבילות מאפשר לזהות דפוסים אופייניים:
  - 1.4.1. **וידאו סטרימינג** ישלח חבילות בגדלים אחידים בהפרשי זמן יציבים.
  - 1.4.2. **שירותי מסרים מיידיים (WhatsApp, Telegram)** יכולו מרווחים בלתי סדירים עם חבילות קטנות יותר.

## עד כמה ניתן לזהות את האתר/אפליקציה?

### 1.5. תרחיש 1: התוקף חשוף לגודל החבילה ולחותמת הזמן בלבד

רמת הדיוק עדיין גבוהה יחסית, שכן תבניות התעבורה ייחודיות לכל שירות.

בעזרת למידת מכונה, תוקפים יכולים לזהות שירותים ספציפיים עבור שירותים כמו YouTube, Netflix, Zoom ו-Spotify, גם בלי לדעת כתובות IP או מספרי פורטים.

מחקרים הראו שניתן לזהות מערכת הפעלה ודפדפן על בסיס מאפייני חבילות בלבד.

### 1.6. תרחיש 2: התעבורה עברה אנונימיזציה (VPN, TOR)

**VPN:** עדיין ניתן לזהות את סוג השירות (למשל, סטרימינג מול שיחת וידאו), אך קשה יותר להבחין בין שירותים דומים.

**TOR:** ההצפנה והתיעול דרך מספר שרתים מטשטשים חלק מהדפוסים, אך תוקף יכול עדיין לשער באיזה סוג שירות מדובר, במיוחד אם נעשה שימוש בחבילות בגודל קבוע או אם יש דפוסים עקביים בין החבילות.

## מה נוכל לעשות כדי להקשות על התוקף?

### 1. שימוש בפאדינג (Padding)

הוספת נתונים רנדומליים לחבילות, כך שכולן יהיו בגודל אחיד (לדוגמה, 1500 בתים), מה שימנע זיהוי של סוג היישום לפי גודל החבילות.

זה מונע מצב לדוגמא שבו תוקף מזהה ששירות סטרימינג משתמש בחבילות גדולות, בעוד שירות VoIP משתמש בחבילות קטנות.

### 2. Traffic Obfuscation (טשטוש תעבורה)

שינוי הדפוסים של שליחת הנתונים, כך שלא יוכלו לזהות אפליקציות על בסיס חותמות זמן.

לדוגמה, שירותי VPN מסוימים ששולחים פקטות דמה או דוחים חבילות בכוונה, כדי לשבש את הדפוסים.

### 3. VPN / Proxy

VPN מצפין את כל התקשורת ושולח את הנתונים דרך שרת ביניים, כך שהתוקף לא יכול לזהות את ה-Tuple-4.

Proxy עובד בצורה דומה אך רק על תעבורה מסוימת (למשל דפדפן).

### 4. שימוש בפרוטוקולים כמו TOR

Tor שולח את התקשורת דרך מספר שרתים שונים בעולם כדי לטשטש את הנתוב.

מכיוון שהחבילות מוצפנות ונשלחות דרך מספר צמתים, תוקף מקומי לא יוכל לראות לאן המשתמש מתחבר.

### 5. הוספת רעש לתעבורה (Traffic Shaping)

שליחת חבילות דמה (Dummy Packets) או עיכוב מכוון של חבילות כדי לבלבל את התוקף.

שימוש בפרוטוקולים כמו FPE (Format-Preserving Encryption) לטשטוש דפוסים.

## בונוס:

שמרנו את ההקלטה לסעיף הבונוס תחת התיקיה `res\pcapfiles\Bonus-Lecture-Mix\` ובנוסף ניתן לגשת גם למפתחות ההצפנה שלה במידת הצורך שנמצאים באותה התיקיה.

כאשר ביצענו את הניסוי מחדש ה- `csv` הראשון שמכיל את כל המאפיינים נשמר ביחד עם המפתחות ולאחר הפענוח בתיקיה :

`res\csv-files\`

וכאשר ביצענו את החלק החלק השני של הניסוי , כלומר התוקף מקבל לידיו רק את גודל הפקטות וחומת הזמן , ה- `csv` נשמר כאשר נמצא בו רק המידע הלא מוצפן (למרות שאין גישה לתוכן לכן זה לא משנה). ניתן למצוא אותו בתיקיה :

`res\csv-files-encrypted\`

### תוצאות מחלק א':

```
=== CLASSIFICATION REPORT ===
              precision    recall  f1-score   support

Audio Streaming      0.90      0.95      0.92        19
  Video Calls        0.95      0.90      0.92        20
Video Streaming      1.00      0.50      0.67         2
  Web Browsing       0.67      1.00      0.80         2

        accuracy                   0.91        43

Accuracy: 0.91
```

### תוצאות מחלק ב':

```
=== CLASSIFICATION REPORT (LIMITED DATA) ===
Class                Precision    Recall    Support
-----
Audio Streaming      1.00      0.00         1
Unknown              0.00      0.00         1
Video Calls          0.00      0.00         1
Video Streaming      1.00      0.00         1
Web Browsing         0.50      1.00         2

Accuracy: 0.33
```



## השוואה בין שתי טבלאות הסיווג

### 1. גודל מערך הדגימות והקטגוריות

בטבלה הראשונה (43 דגימות סך-הכול) מופיעות ארבע קטגוריות (Audio Streaming, Video Calls, Video Streaming, Web Browsing) עם מספר דגימות גדול יחסית בכל קטגוריה. זה מאפשר למודל ללמוד טוב יותר את התכונות הסטטיסטיות של כל סוג תעבורה, ולכן דיוק הסיווג ( $Accuracy=0.91$ ) גבוה יותר. לעומת זאת, בטבלה השנייה (5 דגימות סך-הכול) רואים קטגוריות דוגמת "Audio Streaming", "Video Streaming", "Video Calls", "Unknown", ו-"Web Browsing" — חלקן עם דגימה אחת בלבד. כאשר יש רק דגימה אחת בקטגוריה, דיוק המדדים (Precision, Recall) רגיש לכל טעות בודדת, והתוצאה הסופית נמוכה ( $Accuracy=0.33$ ).

### 2. השפעת מספר הדגימות על ביצועי המודל

כמות גדולה יותר של דגימות בכל קטגוריה (כמו בטבלה הראשונה) מאפשרת למודל להבחין טוב יותר בין סוגי התעבורה. בטבלה השנייה, כל קטגוריה כמעט לא מכילה דגימות, מה שמביא לאי-יציבות במדדים (Precision ו-Recall יכולים להיות 0 או 1 במקרים של הצלחה או כישלון בודד). כתוצאה מכך, הדיוק הכללי נפגע משמעותית.

### 3. הבדלי ביצועים

- בטבלה הראשונה רואים Precision גבוה ( $+0.90$ ) ו-Recall גבוה (0.95, 1.00) ברוב הקטגוריות, משקף יכולת טובה הן לזהות נכון דגימות ששייכות לקטגוריה (Recall) והן להימנע מלסווג בטעות דגימות לקטגוריה לא נכונה (Precision).
- בטבלה השנייה, חלק מהקטגוריות מקבלות  $Recall=0$ , מה שמצביע על כך שדגימה בודדת בקטגוריה לא זוהתה כלל.

### מסקנה מרכזית

כאשר המודל מקבל רק מעט מידע (כמו זמן ואורך החבילה) וכמות דגימות קטנה לכל קטגוריה, הוא מתקשה מאוד להבחין בין פעילויות שונות (כגון ניגון בספוטיפיי לעומת שליחת מייל) ולכן קיטלג אותו בתור Unknown ולא ידע לסווג אותו במקום הנכון ללא מידע נוסף. לעומת זאת, כשהוא מקבל יותר פרטים (כתובות IP, פורטים, מספר זרמים נפרדים בכל CSV) ויכול לדגום כמות דגימות גדולה יותר לכל סוג תעבורה (בגלל ה-flow ID), הוא מסוגל ללמוד חתימה סטטיסטית טובה יותר של כל פעילות ולהגיע לדיוק סיווג גבוה בהרבה.

## נספח 1 : פקודות שנעזרנו באל:

### תסביר לי בקצרה על פרוטוקול mptcp

**MPTCP (Multipath TCP)** הוא הרחבה ל-TCP שמאפשר שימוש בכמה חיבורים בו-זמנית כדי לשפר מהירות, (סלולר + Wi-Fi למשל) יציבות ועמידות בפני ניתוקים. אם חיבור אחד נופל, הנתונים ממשיכים לזרום דרך האחרים.

### כיצד אני בונה בפייתון גרפים?

בפייתון ניתן לבנות גרפים באמצעות הספרייה **Matplotlib**, ולעיתים גם **Seaborn** להדמיות מתקדמות יותר.

### דוגמה בסיסית עם Matplotlib

Copy code

Python

```
import matplotlib.pyplot as plt

# נתונים לדוגמה
x = [1, 2, 3, 4, 5]
y = [10, 20, 15, 25, 30]

# יצירת גרף קווי
plt.plot(x, y, marker='o', linestyle='-', color='b', label="נתונים")

# הוספת כותרות וציר X/Y
plt.xlabel("ציר X")
plt.ylabel("ציר Y")
plt.title("דוגמה לגרף בפייתון")
plt.legend()

# הצגת הגרף
plt.show()
```

תכתוב לי בקצרה - איזה מידע  
חותמת זמן יכולה לתת לתוקף?

חותמת זמן יכולה לחשוף לתוקף:

- **זיהוי אפליקציות** - תבניות זמן אופייניות לכל שירות (למשל, סטרימינג מול צ'אט).
- **מדידת זמן תגובה (RTT)** - מאפשר להסיק על מיקום השרת.
- **שחזור פעולות משתמש** - מעבר בין עמודים, הפעלת שירותים וכו'.
- **התקפות תזמון על הצפנה** - ניתוח זמני שליחה עשוי להסגיר תוכן תקשורת מוצפן.
- **גילוי זהות משתמש אנונימי** - הצלבה עם תעבורה אחרת עלולה לחשוף משתמשי VPN/Tor.

### תסביר לי כיצד שומרים מפתחות tls בקצרה

**שמירת מפתחות TLS** נעשית בדרכים שונות בהתאם לסוג המפתח ולשימוש בו:

1. **מפתחות זמניים (Session Keys)** - נוצרים במהלך ה-TLS Handshake ומשמשים להצפנת התקשורת. לרוב נשמרים בזיכרון (RAM) ונמחקים כשהחיבור נסגר.
2. **מפתחות פרטיים (Private Keys)** - משמשים לזיהוי השרת ונשמרים באופן מאובטח בקובצי מערכת (.pem, .key). ב-HSM (חומרה ייעודית), או ב-TPM (רכיב הצפנה במחשב) כדי למנוע גישה לא מורשית.
3. **מפתחות ציבוריים (Public Keys)** - נמצאים בתעודות TLS (כגון .cert) וניתנים לכל לקוח שמתחבר.

בקצרה, מפתחות TLS זמניים נשמרים בזיכרון ומוחלפים בכל חיבור, בעוד שמפתחות פרטיים מאוחסנים באופן מאובטח להגנה על זהות השרת.

תסביר לי בקצרה כיצד ניתן לבטל שימוש בפרוטוקול quic בגוגל כרום

כדי לבטל את השימוש בפרוטוקול QUIC ב-Google Chrome, בצע את השלבים הבאים:

1. פתח את Chrome והקלד בשורת הכתובת:

chrome://flags/#enable-quic

2. אתר את האפשרות "Experimental QUIC protocol".

3. שנה את הערך מ- "Default" או "Enabled" ל- "Disabled".

4. הפעל מחדש את הדפדפן.

זה ימנע מהדפדפן להשתמש ב-QUIC ויכריח אותו להשתמש ב-TCP במקום.