

# **פרויקט גמר רשתות תקשורת**

**מגישיים:**

אבי נאמן - 318446804

נעוה שלום - 322694530

gil aharon - 209012095

אמנון פוזайлוב - 322275504

קישור לקבצי ה pcap ו�sslkeylogfile הרלוונטיים למטריה.

קישור לfork repository עם קבצי הקוד בgithub.

# **תוכן ענייניים:**

**1. חלק ראשון (תשובות עיוניות)**

**2. חלק שני (תשובות למאמרים)**

CNN+FlowPic •

Early Traffic Classification with ECH •

Analyzing HTTPS Encrypted traffic to IDE •

**3. חלק שלישי (ניתוח הגרפים מהקוד**

ומסקנות)

**4. חלק רביעי (תוצאות הקוד של התוקף**

ומסקנות מהמחקר בחלק זה)

## **תשובות לחלק 1:**

### **1. גורמים להאטת הטעבה:**

#### **בקורת עומס בTCP-**

TCP משתמש במנגנוני בקורת עומס למשל (Slow Start, Congestion Avoidance), כדי למנוע עומס ברשת. אם מתגלה איבוד מנות TCP, מפחית את קצב השידור, מה שעלול להאט את העברת הקבצים.

#### **בעיות בקרה על הזרימה-**

- **גודל חלון מקבל**: אם המקלט מפרוסט חלון קטן בגלל מגבלות בזיכרון הbuffer שלו, זה יאט את הטעבה, גם אם השולח יכול לשולח נתונים בקצב גובה יותר.
- **איבוד מנות ושידור חוזר**: איבוד מנות גורר שידורים חוזרים, מה שגורם לעיכובים. איבוד מנות יכול להיגרם עקב עומס ברשת, חומרה תקולה או שגיאות בקשר.
- **RTT גבוהה**: RTT גבוהה מגדיל את הזמן הדרוש לקבלת אישורים (ACKs) מה שמאט את קצב השידור של השולח.
- **עומס ברשת**: נתבים או מתגים עמוסים עלולים להפיל מנות, מה שגורם למנגנוני בקורת העומס של TCP להפחית את קצב השידור.
- **בעיות בהגדלת חלון TCP**: ברשותות עם רוחב פס גבוה וזמן RTT גבוה, אי שימוש במנגנון TCP Window Scaling עלול להגביל את קצב ההעברה בגלל חלונות קטנים מדי.
- **בעיות MTU ופיצול מנות**: חוסר התאמה - (הגודל המרבי של חבילות MTU או פיצול חבילות, עלולים לגרום לא-יעילות בהעברת הנתונים, במיוחד אם גודל המנות חורג מהמקסימום המותר (MTU) ויש לפצלן).

#### **שלבי פתרון בעיות:**

1. **בדיקה RTT ושיהוי**: ניתן להשתמש בפקודות ping או traceroute כדי לבדוק WHETHERו ולאורך מקטעים ברשת עם זמן תגובה גבוה.
2. **ניתוח גדלי חלונות TCP**: ניתן להשתמש בכלים ניטור רשת כדי לבדוק את גודל חלון TCP- חבילות. אם גודל החלון של המקלט קטן באופן עקבי, ניתן שזה מצביע על צואර בקבוק בצד המקלט.
3. **בדיקה איבוד מנות**: ניתן להשתמש בכלים כמו tcpdump או Wireshark כדי לzechות שידורים חוזרים או ACK כפולים, מה עשוי להצביע על איבוד מנות.
4. **בדיקה אלגוריתמי בקורת עומס**: לוודא שימושם באלגוריתמי בקורת עומס מתאימים (כגון CUBIC בלינוקס), במיוחד ברשותות מהירות.
5. **בדיקה בעיות MTU**: ניתן להשתמש בפקודת ping עם האפשרות `so M`- כדי לבדוק בעיות בזיהוי MTU במסלול הרשת.
6. **nitour עומס רשת**: ניתן להשתמש בכלים ניטור רשת כדי לבדוק ניטור גובה של רוחב פס או איבוד מנות בתבאים ובמוגדים.

## **2. הבנת בקורת הזרימה בTCP:**

בקרת הזרימה נועדה למנוע מהשלוח להציג את המקביל בנסיבות נתונים שהוא אינו יכול לעמוד. מנגנון זה מתבצע באמצעות חלון TCP- של המקביל, (window) הקובע כמה נתונים לא מאושרים יכולים להיות בתעבורה בכלל רגע נתון.

### **השפעה על הביצועים כאשר השולח מהיר יותר מהמקבל:**

- **צואר בקבוק מצד המקבל:** אם השולח בעל כוח עיבוד גבוה משמעותית יוכל לשЛОח נתונים מהר יותר ממה שהמקבל יכול לעמוד, חלון TCP-של המקביל יתמלא לתמלא בנסיבות. כאשר זה קורה, השולח ייאלץ להשנות את ההעברה עד שיתפנה מקום בזיכרון החוץ של המקביל.
- **תקופות השבתה של השולח:** השולח יכנס לעתים קרובות למצבים של המתנה לקבלת אישורים (ACKs) שמעידים כי המקביל מוכן לקבל עוד נתונים, מצב זה גורם לניצול לאיעיל של משאבי השולח.
- **ניתול לא אופטימלי של רוחב הפס:** גם אם השולח יכול להעביר נתונים בקצב גבוה יותר, בקרת הזרימה תגביל את קצב העברה כך שיוכלה ליכולה של המקביל, מה שעולול לגרום לביצועים נמוכים מהפוטנציאלי.
- **השיigkeit גבירות בהעברת קבצים גדולים:** בהעברת קבצים גדולים, בקרת הזרימה עשויה להאריך את זמן העברה, לאחר שהשידור צריך להסתגל באופן רציף למהירות של המקביל.

### **\*דריכים להקטנת הפגיעה בביצועים:**

- **шиיפור ביצועי המקביל:** הגדלת גודל החוץ (buffer) מצד המקביל או שיפור יכולות העיבוד שלו כדי לאפשר עיבוד יעיל יותר של הנתונים הנכנסים.
- **הגדלת חלון TCP :** הפעלת Window ScalingTCP תאפשר חלונות גדולים יותר, מה שיצמצם את תדרות ההשנות הנגרמות מבקרת הזרימה.
- **שימוש בחיבורים מקבילים:** פיתוח מספר חיבור TCP במקביל יכולה לשפר את נגישות המשאבי של השולח, במיוחד במקרים של העברת קבצים או סטרימיניג.

### **3. כיצד בחרת מסלול משפיעה על ביצועי הרשת:**

- **שיiei ועיכובים:** בחרית המסלול משפיעה על זמן הסביב (RTT), מסלולים קצרים יותר או פחות עמוסים ממחזירים שהו, מה שմשפר את זמן התגובה של הרשת.
- **ניתול רוחב פס:** מסלולים מסוימים מציעים קיבולת רוחב פס גבוהה יותר, מה שמאפשר העברת נתונים מהירה יותר, בחרית מסלול עם רוחב פס נמוך עלולה ליצור צוארי בקבוק.
- **אמינותות וגיבוי:** פרוטוקולי ניתוב לוחכים לחישוב את יציבות המסלול, מסלול יציב עם פחת תקלות משפר את ביצועי הרשת ומפחית סיכון לניטוקים.
- **עומס ואיזון עומס:** החלטות ניתוב יכולות לפזר תנועה בין מסלולים שונים כדי למנוע עמוסים ולשפר ביצועים. ניתוב לא מאוזן עלול להעמיס על מסלולים מסוימים תוך השארת אחרים בתת-ניתול.

### **גורםים שיש לקחן בחשבון בהחלטות ניתוב:**

- **עלות קישור/מדדים:** פרוטוקולי ניתוב משתמשים במדדים כמו מספר קפיצות (hop count) שהו, רוחב פס ואמינותות כדי לקבוע את המסלול האופטימלי.
- **פרוטוקולי ניתוב (OSPF, BGP):** Open Shortest Path First (OSPF) (Border Gateway Protocol) משמשים באלגוריתמים שונים לקבלת החלטות ניתוב.
- **aicoot שירות(SoQ):** מסלולים מסוימים עושים להעדיף סוג תעבורה מסוימים (כגון שיחות ו叫声) או שירותים מסוימים (כדי להבטיח איכות השירות גבוהה).

- **인터넷 זולים יותר או מסלולים מאובטחים יותר.**

## 4. מהו MPTCP?

(MPTCP) Multic平路 TCP מאפשר לחברו TCP יחד לשמש במספר מסלולים בו-זמנית, תוך שימוש רבוי משקי רשות (כגון Wi-Fi, Ethernet, LTE) כדי לשפר תפקוה ואמינות.

### יתרונות של MPTCP:

- **הגדלת תפקוה:** על ידי שילוב של מספר משקי רשות MPTCP, מושג תפקה מצורפת גבוהה יותר בהשוואה לTCPרגיל המשמש במסלול אחד.
- **אמינות וגיבוי משופרים:** אם מסלול אחד נכשל MPTCP, מעביר את התעבורה באופן שקווי למסלול חלופי ללא ניתוק החיבור.
- **אייזן עומסים עיל:** MPTCP מפזר את הנתונים דינמית בין מספר מסלולים בהתאם לעומס הרשות ולאיכות המסלול, לשימוש אופטימלי במשאבים.
- **הפחתת��יהן:** ניתן לבחור את המסלול המהיר ביותר עבור זרמי נתונים מסוימים, מה שמחמיא את זמן התגובה ומשפר את ביצועי הרשות.
- **תמייה טוביה יותר בנויות:** אידיאלי למקרים נידים שעוברים מרשת Wi-Fi לרשות סלולרית מבלי לאבד את החיבור.

## 5. גורמים אפשריים לאיבוד מנות בשכבות הרשות:

- **עומס בנתביים:** נתבים עמוסים עלולים להפעיל מנות כאשר התוריים שלהם מתמלאים.
- **חומרה תקולה:** כבילים פגומים, נתבים או מתגים תקולים עשויים לגרום לשגיאות ולמנות פגומות.
- **lolאות ניתוב או תצורה שגיה:** נתבים שהוגדרו בצורה שגיה עלולים לגרום למנות להסתובב ללא סוף עד שהן נזקקות.
- **חווסף התאמה בסלט:** בעיות בסלט-עלולות לגרום לפיצול מנות או להפלtan אם התקנים באמצעות המסלול אינם יכולים להתמודד עם גודל מנות גדול.

### גורמים אפשריים בשכבות התעבורה:

בקרט עמוס TCP- איבוד מנות מפעיל את מגנוני בקרת העמוס של TCP מה שמחמיא את קצב השידור של השולח.

גודש בזכרון חוץ (Buffer Overflows): חוסר מקום בחוץ של השולח או המקלט עלול לגרום להפלת מנות.

זמן המתנה ושידורים חוזרים: שיוי גובה או תנודות ברשות (jitter) עלולים לגרום לפקיעות זמן (timeouts) שימושיים לשידורים חוזרים ולתפיסה של איבוד מנות.

### צעדים לפתרון בעיות איבוד מנות:

1. **נטור עמוס ברשות:** ניתן להשתמש בכלים כמו netstat או Wireshark כדי לאתר קישורים עמוסים ולהחליק מחדש את התעבורה.
2. **בדיקה והחלפת חומרה תקולה:** ניתן לבדוק ולהחליף כלים, מתגים או נתבים תקולים לפי הצורך.

3. **אופטימיזציה של תצורת ניתוב**: לוודא שהגדירות הניתוב נכונות ואין לולאות ניתוב.
4. **התאמת הגדרות MTU**: ניתן להשתמש בזיהוי MTU במסלול (Path MTU Discovery) כדי לקבוע גודל MTU מתאים ולהימנע מפיצול מנות.
5. **הגדלת חיצבי זיכרון (Buffer Sizes)**: בנתב או בנקודות הקצה, ניתן להגדיל את קיבולת החיצץ כדי להתמודד עם גלי תנועה פתאומיים.
6. **הפעלת מדיניות QoS**: ניתן לתעדף תעבורת קריטית כגון VoIP כדי להבטיח משLOW אמין בזמן עומס.

## חלק 2 (ניתוח המאמרים):

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

#### **תרומות עיקריות של המאמר:**

המאמר מציג גישה חדשה לסיווג תעבורת אינטרנט מוצפנת באמצעות טכניקות למידת מכונה המשמשות בדרך כלל בזיהוי תמונות. התורמות המרכזיות כוללות:

- יצוג FlowPic: המרת נתוני זרם תעבורת לרשות (network flow)לייצוג תמנוני בשם FlowPic אשר לווד מידע על גודל המנות (packets) וזמן הגעה שלהם.
- שימוש CNNs ל识别 תעבורת: שימוש רשתות ניירונים (CNNs) ל识别 תעבורת, במקום הסתמכו על מאפיינים סטטיסטיים מוחלצים ידנית.
- דיוק סיווג גבוה: השגת למעלה מ-96% דיוק בזיהוי קטגוריות תעבורת, כולל תעבורת מוצפנת VPN וTOR.
- יכולת הכללה לשימושים חדשים: הוכחת יכולת המודל ל识别 בהצלחה שימושים שלא נראה במהלך האימון, עם דיוק של 99.9%.
- שמור פרטיות: סיווג תעבורת ללא שימוש בתוכן עצמו ובכך הימנע מהפגיעה בפרטיות.

#### **מאפייני התעborת והשיפורים הייחודיים:**

המאמר מציג את FlowPic אשר מבוסס על:

- גודל מנות
- זמן הצעת מנות
- התפלגות מנות לאורך זמן: ייצוג כHistogram 2D.

#### חדשניים עיקריים:

- ייצוג תמנוני של זרם תעבורת: במקום שימוש במאפיינים סטטיסטיים מסורתיים, השיטה יוצרת תמונה (היסטוגרמה דו ממדית) המבוססת על גדלי מנות וזמן הגעה, ומאפשרת ל-CNN ל识别 את התעborת בצורה ויזואלית.
- חווסף תלות בתוכן החבילה: בניית שיטות מסורתיות כמו DPI (Deep Packet Inspection), FlowPic וכו' לאיננו מסתמך על תוכן המטען, ולכן הוא אפקטיבי גם עבור תעborת מוצפנת.
- ניתוח חלון זמן קצר: המערכת מסוגלת ל识别 זרמי תעborת קצרים בכיוון אחד בלבד, ללא צורך בניהו זרם דו-כיווני מלא.

## **תוצאות עיקריות ותובנות:**

### **דיק בסיג:**

דיק	משימה
85.0%	סיג תעבורת לא-VPN
98.4%	סיג תעבורת מוצפנת בVPN-.
67.8%	סיג תעבורת מוצפנת בSSL/TLS-.
99.4% - 78.9%	סיג תעבורת מוצפנת בVPN- כאשר האימון בוצע על נתונים ללא VPN
99.7%	זיהוי יישומים

### **תובנות מרכזיות:**

- שיטת FlowPic המבוססת על CNNs מSIGA ביצועים טובים יותר בהשוואה לשיטות מסורתיות של למידת מכונה.
- המודל מזהה קטגוריות תעבורת גם כאשר נעשה שימוש בהצפנה כמו VPN או TOR
- גם כאשר מסרים יישומים מסוימים מהאימון, המודל עדין מצליח לסוג אותם נכון, מה שਮעיד על חווון גבוה.
- השיטה מציגה דיק גבוה משמעותית בהשוואה למחקרים קודמים, במיוחד בזיהוי יישומים.

### **איורים מרכזיים ותובנות חזותיות:**

1. דוגמאות FlowPics ליישומי יידאו: מציג כיצד יישומים שונים כגון Netflix ו-Skype יוצרים תובנות תעבורת ייחודיות.
2. השוואת קטגוריות תעבורת תחת הצפנות שונות: מדגים כיצד תעבורת IPoV וצ'אט משתנה כאשר נעשה שימוש בVPN או TOR
3. ארכיטקטורת הNN - בהשראת LeNet-5: איור של שכבות הקונבולוציה המשמשות לסיג FlowPic
4. מטריצות בלבול: השוואת דיק הסיג בין תעבורת רגילה, TOR, VPN, FlowPic

### **הסבר האלגוריתם המוצג במאמר למען ההקשר (flowpic+CNN):**

המאמר מציין שימוש ב-CNNs-טכניקה נפוצה בזיהוי תמונות – לסיג תעבורת אינטרנט מוצפנת. הרעיון המרכזי הוא שתבניות תעבורת ניתנות לייצוג כתרומות (FlowPics), מה שמאפשר לNN להזות ולסיג אותן באופן דומה לזיהוי אובייקטים בתמונות.

כיצד CNNs משמשים בסיווג FlowPics

1. המרת נתונים הרשות לFlowPics

- גודל המנות וזמן ההגעה מתרגמים לתמונה בגודל  $1500 \times 1500$  פיקסלים בגוני אפור.
- ציר ה-X מייצג את זמן ההגעה, וציר ה-Y מייצג את גודל המנה.
- עוצמת הפיקסלים משקפת את מספר המנות עם גודל וזמן מסוים.

2. העברת FlowPic למודל CNN

- הNN מזהה תבניות בתמונה ומסווג את סוג התעבורה.

3. שימוש בNN-CNN-לסיווג:

- המודל מסווג את הFlowPic-לקטגוריות כמו VoIP וידאו, צ'אט, העברת קבצים וגלישה.

הסבר על מבנה האלגוריתם:

המודל מבוסס על ארכיטקטורת LeNet-5 אשר תוכננה במקור לדיזייני ספרות בכתב יד, עם התאמות לסיווג תעבורה.

מבנה הרשות:

- שכבת קלט FlowPic בגודל  $1500 \times 1500$  מזון ל-CNN-.
- שכבות קונולוציה:
  - CONV1: 10 מסננים בגודל  $10 \times 10$  עם צעד של 5, מפיק 10 מפות מאפיינים בגודל  $300 \times 300$ .
  - Max Pooling, מפחיתה את הגודל ל- $150 \times 150$ .
- CONV2: 20 מסננים בגודל  $10 \times 10$  עם צעד של 5, מפיק 20 מפות מאפיינים בגודל  $30 \times 30$ .
  - Max Pooling מפחיתה את הגודל ל- $15 \times 15$ .
- שכבת Fully Connected עם 64 נוירונים.
- שכבת פלט עם softmax (פונקציית הפעלה אשר ממירה את ערכי הפלט לווקטור של הסתברויות שמסתכמת ל-1). לסיווג קטגוריות התעבורה.

יתרונות השימוש בNN-CNN-בסיווג תעבורה מוצפנת:

- לומד מאפיינים אוטומטית, ללא צורך בהגדרת מאפיינים ידנית.
- אפשר סיווג תעבורה גם כשהיא מוצפנת.
- משיג ביצועים גבוהים יותר מאשר שיטות מסורתיות כמו SVM, KNN,

- מסוגל להקליל ולזהות יישומים שלא נראו במהלך האימון.

#### **מסקנות עיקריות מהמאמר:**

1. שיטת חדשנית ומדויקת לסוג תעבורה מוצפנת: השיטה מצילהה לסוג קטגוריות תעבורה (ג'ילישה, צ'אט, וידאו, VoIP, העברת קבצים) בדיק שמעל 96%, תעבורה שעוברת דרך VPN מסופגת בדיק של 99.2%, ודרך Tor בדיק של מעל 89%, למעט העברת קבצים.
2. יכולת הכללה על אפליקציות חדשות: גם כאשר האפליקציה הספציפית לא הופיעה באימון, ניתן עדין לסוג את התעבורה לקטגוריה הנכונה, למשל, המודל אומן ללא נתונים Facebook Video אך הצליח לזהות אותו כוידיאו בדיק של 99.9%.
3. עדיפות על שיטות קודמות: השיטה ייעילה יותר מהשיטות הקלאסיות של למידה מפוקחת המשמשות בתוכנות מהונדסות כגון KNN, SVM בשונה מניתוח חבילות عمוק (DPI) השיטה אינה פולשנית לפרטיות ומتبוססת רק על גודל וזמן ההגעה של החבילות.
4. התמודדות עם הצפנה(Tor,VPN): השיטה מצילהה לסוג תעבורה למטרות הצפנה, בזכות שימוש במידע זרימה בלבד, זיהוי סוג הצפנה מתבצע בדיק של 88.4%, כאשר Tor מזווה בדיק של 97.7%.
5. פשטות ויעילות של מודל CNNC: הארכיטקטורה מבוססת LeNet-5 עם שכבות קונבולוציה, מילוק (Max Pooling) ושכבות Fully Connected, השיטה פועלת ללא התאמות פרטניות לכל סוג תעבורה, מה שמאפשר שימוש חזזר באוטה ארכיטקטורה לכל משימות הסיווג.
6. אפשרויות לשיפור עתידי: ניתן לשפר את ביצועי המודל ע"י התאמות בהיפר-פרמטרים, למשל הקטנת רזולוציית FlowPic מ- $1500 \times 1500$  ל- $300 \times 300$ , ניתן לשלב הצבעה מבוזרת (Voting) בין מספר חלונות זמן כדי לשפר דיוק.

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

### **תרומות עיקריות של המאמר:**

המאמר מתמקד בסיווג מוקדם של תעבורת רשת (eTC) במיוחד בהקשר של הצפנה TLS 1.3 ושימוש ב (ECH) Encrypted ClientHello, התרומות המרכזיות כוללות:

- מאגר נתונים רחב-היקף חדש: המאמר מציג מאגר נתונים מגוון הכולל יותר מ- 600,000 זרמי TLS שנאספו מאזרחים שונים בעולם (צפון אמריקה, אירופה ואסיה). מאגר זה הוא אחד ממאגרי הנתונים הפומביים הגדולים ביותר לסיווג תעבורה.
- מסoug תעבורה היברידי מבוסס (hRFTC) : המאמר מציג מסoug חדשני המשלב מטא-נתונים לא מוצפנים של TLS key share, כגון רישימת הצפנים והרחבות עם מאפיינים סטטיסטיים מבוסס זרם (כגון זמני הגעה בין מננות והתפלגות גDALI מננות).
- הערצת שיטות קיימות לסיווג תעבורה: המאמר משווה מסougים מתקדמים (מבוסס מננות ומבוסס זרמים) על תעבורה מוצפנת ב, Hello-Client ומחה את חולשותיהם ומצביע שיפורים.
- תובנות על שונות גאוגרפיה: המאמר מראה כי מסougים שאומנו באזורי גאוגרפיה מסוימים אינם מתקפקדים היטב באזורי אחרים, ודורשים אימון מחדש כדי להציג לביצועים אופטימליים.

### **מאפייני התעבורה והשיפורים הייחודיים:**

המסoug החדשני hRFTC שמשלב:

- מאפיינים מבוסס מננות: שדות הקשרים לשלב ההתחלתי של חיבור TLS כגון רישומות הצפנים והרחבות.
- מאפיינים מבוסס זרם: סטטיסטיות של גודל מננות, זמני הגעה של מננות, וכוכנות מבוססות רצפים.

### חידושים עיקריים

- שימוש במטא-נתונים של TLS לסיווג: למרות שהצפנה HEC מסתירה חלק מהמידע, עדין קיימים שדות בלתי מוצפנים שיכולים לשמש לסיווג. המסoug מנצל שדות אלו בצורה אופטימלית.
- מאפיינים מבוסס זרם עבור סיווג מוקדם: במקום להמתין לקבלת אלף מננות (כפי שנדרש בשיטות מסורתיות) hRFTC, מצליח לסoug תעבורה באמצעות מספר קטן של מננות ראשונות בלבד.
- שילוב היברידי לשיפור הדיוק: ההילוב בין מאפיינים מבוסס מננות וזרים משפר משמעותית את הביצועים בהשוואה למסougים טהורים המבוססים על אחת מהשיטות בלבד.

## **תוצאות עיקריות ותובנות:**

### **ביצועי שיטות סיווג שונות:**

Macro F-Score	שיטת
94.6%	hRFTC השיטה המוצעת
86.3%	University of Waterloo מודל
79.4%	C4.5 מושג מבודוס
72.1%	CESNET זרם מבוסס
38.4%	MATEC רשת נוירונים מודל
35.7%	GRU-U מנגנון תשומת לב רשת GRU
39.2%	RB-RF Random Forest מושג מנות

### **מסקנות עיקריות:**

- hRFTC מושג ביצועים טובים יותר מכל שאר המושגים, כולל רשתות נוירונים מתקדמות.
- מושגים מושג מנות נכשלים (דיק ש 38.4%) כאשר נעשה שימוש בהצפנה ECH, אך שילובם עם מאפיינים מושג זרם מאפשר להגעה לדיק ש 94.6%.
- מודלים שנאומנו באזורי אחד אינם מתפקדים היטב באזורי אחר, בשל הבדלים בהגדרות TLS ודף עבורה.

### **הסבר על האלגוריתם המדבר במאמר למן ההקשר (hrftc):**

hRFTC (Hybrid Random Forest Traffic Classifier) הוא אלגוריתם מתקדם לסיווג מוקדם של תעבורה, (eTC) המשלב בין מאפיינים מושג מנות זרם כדי להתמודד עם הצפנה (ECH) Encrypted ClientHello (ECH).

### **שלבי העבודה של hRFTC**

#### **שלב 1: חילוץ מאפיינים**

- חילוץ מטא-נתונים בלתי מוצפנים מתוך ClientHello ו-ServerHello.
- חישוב מאפיינים סטטיסטיים מהתנהגות התעבורה בשלבים הראשונים של החיבור.

#### **שלב 2: שילוב מאפיינים**

- **שילוב של נתונים מבוסס<sup>i</sup> מנות עם נתונים מבוסס<sup>i</sup> זרם לווקטור מאפיינים אחד.**
- **נرمול המאפיינים והתמודדות עם נתונים חסרים.**

### **שלב 3: סיווג עRandom Forest**

- **שימוש באלגוריתם Random Forest לשסיווג תעובה בזמן אמת.**

#### **יתרונות השימוש בRandom Forest-Random Forest-CNN-**

- **עמידות לרעש: בניגוד לרשתות ניורוניים, המודל אינו רגיש יתר על המידה לשינויים קטנים ברכפי מנות.**
- **ביצועים מהירים: המסואג יכול לעבוד 100,000 זרמים בשניה, מה שהופך אותו למתאים לשימוש בזמן אמת.**
- **יכולת זיהוי של קשרים מורכבים בין מאפיינים מבוסס<sup>i</sup> מנות וזרם.**

## מאמר :Analyzing HTTPS Encrypted Traffic to IDE

המאמר מציג שיטה חדשה לזיהוי מערכת הפעלה, הדפסן והישום של משתמש על בסיס ניתוח תעבורת HTTPS מוצפנת בלבד.

### **התרומות המרכזיות הן:**

- המחקר הראשון לסייע מערכת הפעלה, דפסן ויישום מתעborת HTTPS : מראה כי** יRib פסיבי יכול להסיק את מערכת הפעלה, הדפסן וסוג היישום למרות הצפנה HTTPS, מדגים כי ניתן לבצע סיווג על סמך תכונות שמופקות ממטא-נתונים של TLS/SSL והתנהגות רשותית.
- ט' תכונות חדש לשיפור הסיווג:** מציע סט חדש של תכונות הקשורות להתחנאות "מתפרצת" בתעבורה (פיקים של פעילות), מציג תכונות של לחיצת יד TLS/SSL/TLS.
- האפשרויות הבינה בין דפסנים ויישומים,** משיג דיוק של 96.06% בסיווג, שיפור לעמודת 93.51% בבסיס.
- בסיס נתונים רחב היקף:** מספק בסיס נתונים של מעל 20,000 שנים מסומנים, הכוללים סוגים שונים של מערכות הפעלה (וינדוס, לינוקס-אובונטו, macOS, macOS) דפסנים (כרום, פירפוקס, ספארי, אקספלורר) ויישומים (ווטוב, פיסבוק, טויטר) מסד הנתונים פתוח למחקר נוסף.
- מודל סיווג מדויק במיוחד:** משתמש במכונת קטררים תומכת (SVM) עם קرنל RBF, מעריך מספר סטימ של תכונות ומראה כי שילוב התכונות הבסיסיות עם התכונות החדשנות נותן את התוצאה הטובה ביותר.

### **תכונות תעבורה בשימוש:**

#### 1. תכונות בסיסיות (ນפוצות בסיווג תעבורה):

##### **תכונות מבוססות חבילות**

- מספר החבילות שנשלחו קידמה/אחריה.
- סך כל הבטים שנשלחו/התקבלו.
- זמן הגעה מינימליים, מקסימליים וממוצעים.
- סטיית תקן של גודל החבילה ושל זמן ההגעה.
- סך כל מספר החבילות, יחס הכווניות, וערך TTL

### **תכונות חדשות שהוצגו במאמר**

#### **תכונות חדשות שמשפרות את הדיוק של הסיווג:**

- תכונות SSL/TLS:**
  - גרסת ה TLS-בשימוש.
  - מספר ההרחבות ב TLS handshake.
  - אורך מזהה הסשן של SSL.

- מספר שיטות ההצפנה של SSL
- תכונות התנהגות "מתפרצת":

  - מדידת "פיקם" בתעבורה – התפרציות של נתונים עם תקופות שקט ביןיהן.
  - זיהוי יישומים שונים לפי מבנה התעבורה שלהם, משמש להבדלה בין דפדףים לפי דפוסי הפיקים הייחודיים שלהם.

השילוב של תכונות אלו מאפשר למודל להבדיל בין דפדףים ומערכות הפעלה בעלות דפוסי תעבורה דומים.

### **תוצאות עיקריות ותובנות**

#### דיקן הסיווג

סט תכונות	דיקון סיווג יישום	דיקון סיווג דפדף	דיקון סיווג הפעלה
<b>תכונות בסיסיות</b>			
92.3%	93.51%	93.52%	
93.7%	94.2%	94.5%	<b>תכונות חדשות בלבד</b>
96.06%	96.06%	96.06%	<b>שילוב בסיס + תכונות חדשות</b>

#### נתונים חשובים ותובנות ויזואליות

#### **הרכב מסד הנתונים**

- **מערכות הפעלה:** ווינדוס, אובונטו, OS Mac , אוביון
- **דפדים:** כרום, פירפוקס, ספארי, אינטרנט אקספלורר.
- **ישומים:** יוטיוב, פיסבוק, טוויטר, דרופובוקס , TeamViewer

#### **חשיבות תכונות**

- תכונות SSL מסוימות להבחין בין דפדים וסוגי מערכות הפעלה.
- תכונות מבוססות זמנים של חבילות מושפרות את סיווג היישומים.

#### **גרף שיפור הדיקון**

- **שילוב בסיס בלבד** משיגות דיקון של ~93%.
- **הוספה** **תכונות פיקם** ומטא-נתונים של SSL מושפרת את הדיקון ל-96.06%.

## **מסקנות עיקריות מהמאמר**

- אם אמ' הנתונים מוצפנים ב-HTTPS- ניתן להסיק מידע על המשתמש
  - ניתן לזהות את מערכת הפעלה, הדפדפן והישום באמצעות ניטוח מטא-נתונים של TLS ומבנה התעבורה.
  - תוקפים יכולים לעקב אחר משתמשים ולזהות פעילותם גם ללא גישה לנ נתונים עצם.
- תוספת התכונות חדשות משפרת את דיווק הסיווג
  - שימוש בתכונות מבוססות TLS והתנהגות מתפרצת (bursty behavior) מעלה את דיווק הסיווג ל-96.06%.
  - שילוב נתוני TLS וזמן חבילות מס'יע בדיהו' מדויק של הדפדפן והישום.
- ניתן לזהות דפנינים ויישומים לפי דפוסי שליחת הנתונים שלהם
  - דפנינים שונים (Chrome, Firefox, Safari) יוצרים דפוסי זרימה ייחודיים בראשת.
  - אתרים כמו YouTube או Facebook-שולחים נתונים באופן שמאפשר להבדיל ביניהם.
- השלכות על פרטיות – ניתן לזהות משתמשים גם ללא חשיפת הנתונים
  - תוקפים יכולים להשתמש במידע זה כדי לעקב אחריו משתמשים ולבצע התקפות ממוקדות.
  - רשותם פרטום יכולות לנצל זאת כדי לזהות דפנינים ויישומים לצורך מעקב והתאמת פרסומות.
- כיצד ניתן לשפר פרטיות?
  - שימוש בVPN-ובהשhiveightabilות עשו למונע זיהוי המבוסס על דפוסי תעבורה.
  - פרוטוקולים כמו (Hello Client Encrypted) ECH יכולים להצפין מידע מטא- данה ולמנוע זיהוי של הדפדפן והאפליקציה.

## חלק 3 (ניתוח הגרפים ומסקנות):

### טבלת המידע המיצגת את קבצי pcap השוניים:

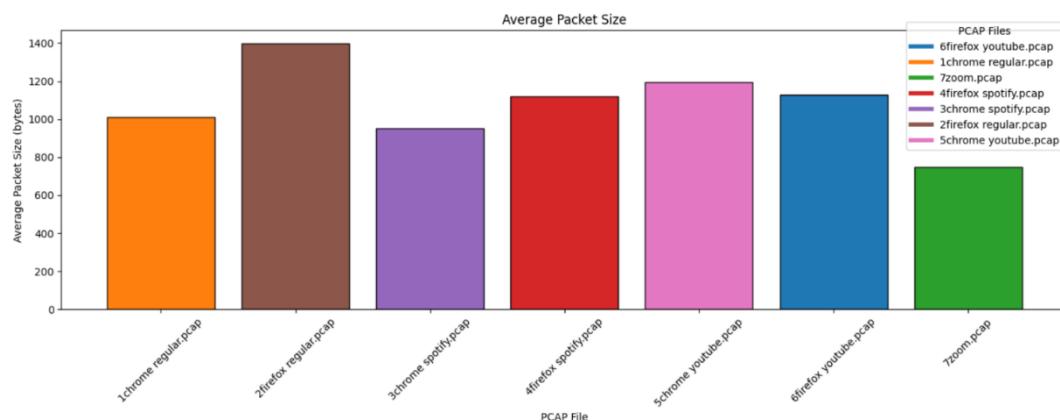
Pcap file	Flow size	Flow Volume (bytes)	Flow duration (second)	Avg Packet size (bytes)	Avg Packet IAT (second)	CV IAT	Unique Flows	Flow Directionality Ratio	Http Count	Tcp Flags	Ip protocols
1chrome regular.pcap	32596	32973434	178.73	1011.58	0.005483	21.770511	647	0.79	HTTP2-4646 HTTP3-4C SYN-713 ACK-14929 P-6-15481 17-17077 1-10		
2firefox regular.pcap	28912	40374111	184.89	1396.45	0.006395	12.682749	643	0.754	HTTP2-5115 HTTP3-2C ACK-19071 PSH-9140 S-6-20262 17-8536 1-90		
3chrome spotify.pcap	8637	8236517	186.59	953.63	0.021603	15.310512	169	0.27	HTTP2-90 HTTP3-59 ACK-1153 FIN-70 SYN-17-7381 1-15 6-486		
4firefox spotify.pcap	10978	12288540	186.83	119.38	0.017018	13.173167	203	0.231	HTTP3-364 HTTP1-12 ACK-2210 PSH-747 FIN-6-2279 17-8664		
5chrome youtube.pcap	22364	26672713	184.32	1192.66	0.008242	22.128857	86	0.156	HTTP2-311 HTTP3-85 ACK-1134 SYN-36 PSH-17-21178 1-11 6-10		
6firefox youtube.pcap	19841	22406295	188.96	1129.29	0.009524	15.399508	96	0.116	HTTP1-18 HTTP3-391 SYN-40 ACK-584 PSH-6-964 17-18827		
7zoom.pcap	25433	19037268	180.35	748.53	0.007091	1.80702	33	0.558	0 ACK-2117 PSH-1087 FI-17-23307 6-2120		

### הציג הגרפים מהקווד איתם השווינו בין התעבורות השונות :

קודם כל למען ההשוואה נציין, כי לכל ההקלטות המתועדות כאן זמן ההקלטה הוא בקירוב 3 דקות, שבהן נרמל את ההשוואה בין כל התעבורות על גבי זמן, וגם כדי שנוכל לזהות דפואים מסוימים שבקלטות קצרות יותר יכול להיות והיו מתפספים.

#### 1. גודל חביליה ממוצעת: Average Packet Size.

הגרף מציג את גודל החביליה הממוצע עבור קבצי PCAP שנאספו, ומאפשר לזהות דפואים ברורים בהתקנות התעבורה של יישומים שונים.



#### מסקנות מהגרף:

##### 1. גודל חביליה ממוצע של firefox גדול מchromes.

- הגורם לכך הוא FirefoxC נוטה לבצע פחות פיצול של חבילות TCP לעומת chrome.
- מנגנון ניהול TCP של firefox מאחד יותר נתונים לפני שליחת חביליה, בעוד chrome ששולח חבילות קטנות יותר באופן תדייר.
- Firefox מבצע שליחה מבוזרת ומהירה יותר של חבילות קטנות, בעוד chrome ממקסם את גודל החביליה (MTU) במקרים מסוימים, מה שוביל לגדיל חביליה ממוצע גבוהה יותר.

##### 2. גודל חביליה ממוצע של zoom קטן מכל יתר הגלישות.

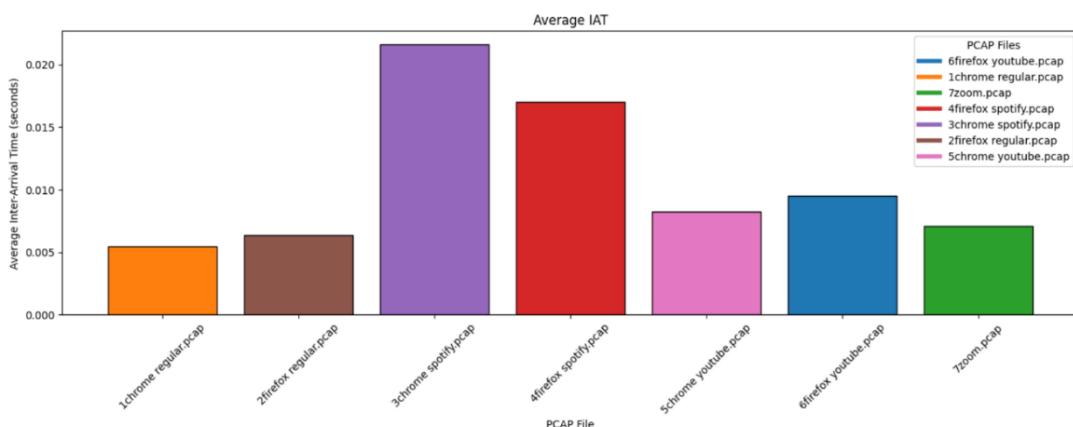
- Zoom משתמש בקפס (במקום TCP) להעברת חבילות מדיה בזמן אמיתי, מה שמקטין את הצורך באיחוד נתונים ושלוח חבילות קטנות בקצב מהיר.
  - פרוטוקול (RTP) (Real-time Transport Protocol) Zoom משמש את Zoom לבני לשלוח מידע בתדריות גבוהה מאוד אך עם חבילות קטנות, כדי למנוע השהיות במקרה מקבל.
  - גודל חבילה קטן מסייע להימנע מתקוני שגיאות כבדים שעולים לגורם לעיכוב בשידור.
- 3. גודל חבילה ממוצעים של YouTube-Spotify-Dומים מאוד בין שני הדפדפניים.**
- YouTube-Spotify-משתמשים בפרוטוקול HTTP מבוססי DASH/HLS, המאפשרים הורדת מקטע מדיה גדולים בגודל קבוע פחות או יותר.
  - בשני הדפדפניים, תובורת המדיה מגיעה מהשירותים באותו אופן, והגודל הממוצע של חבילות הנתונים נשמר בסטיה מינימלית.
  - שני השירותים משתמשים ב-adaptive bitrate streaming buffering וב- caching החבילות שלהם דומות מבחינה הגודל הממוצע.

## 2. זמן הגעה ממוצע בין חבילות IAT – Average Inter-Arrival Time

הגרף מציג את ה-IAT-הממוצע, כלומר הזמן הממוצע בין הגעת חבילה אחת לחבילה הבאה שנקלטה במהלך תüberות הרשת של כל אחת מהאפליקציות/הדפדפניים שנבדקו.

זמן הגעת חבילה ממוצע חשוב מכיוון שהוא מספק תובנות לגבי אופן העברת הנתונים של "ישומים שונים":

- IAT נמוך מעיד על שליחה רציפה וצפופה של חבילות נתונים.
- IAT גבוהה מצביע על שליחה מפוזרת יותר, כלומר יש מרוחך זמן גדולים יותר בין חבילה לחבילה.



## **מסקנות מהגרף:**

### IAT ממוצע כמעט זהה בגלישת דף דין רגילה בין שני הדפדףים.

- ניתן לראות מהגרף כי (regular) Firefox ו-Chrome מציגים ערכי IAT דומים מאוד. הסיבה לכך היא ששני הדפדףים מבצעים בקשות HTTP באופן דומה, בהתבסס על פרוטוקול TCP, שבו מנגנון הזרימה של הנתונים עובדים בצורה קרובת מאוד.

- TCP מבצע ניהול Congestion Control דומה בשני הדפדףים, כך שהמרווח בין החבילות נשאר יציב.

### IAT הממוצע של Spotify גדול משמעותית בהשוואה לשאר הגלישות.

- Spotify (בשני הדפדףים) מציג IAT גבוה מאוד בהשוואה לשאר היישומים. הסיבה לכך היא שהזרמת מוזיקה ב-Spotify משתמשת ב-buffering האפליקציה מורידה מקטע אודיו גדולים מראש (pre-fetching) וכך אין צורך בשילוחה תקופה של חבילות נתוניות.
- בכלל buffering, חבילות גדולות נשלחות בפרק זמן יחסית רחוקים זה מזה, מה שגורם לעלייה ב-IAT.

### IAT הממוצע של יוטיובBINONI לשאר הגלישות

- YouTube משתמש בפרוטוקול DASH (Dynamic Adaptive Streaming over HTTP) שבו החבילות מגיעות בקצבות קטנות בקצב קבוע יחסית.
- הסיבה לכך היא שYouTube מבצע התאמת קצב הזרמה (bitrate adaptation) בהתאם לרוחב הפס, כך שהשידור חלק, אך אין הצפה של חבילות כמו בזום או בגלישה רגילה.

### IAT ממוצע של זום והדפדףים קטן יחסית לשאר הגלישות:

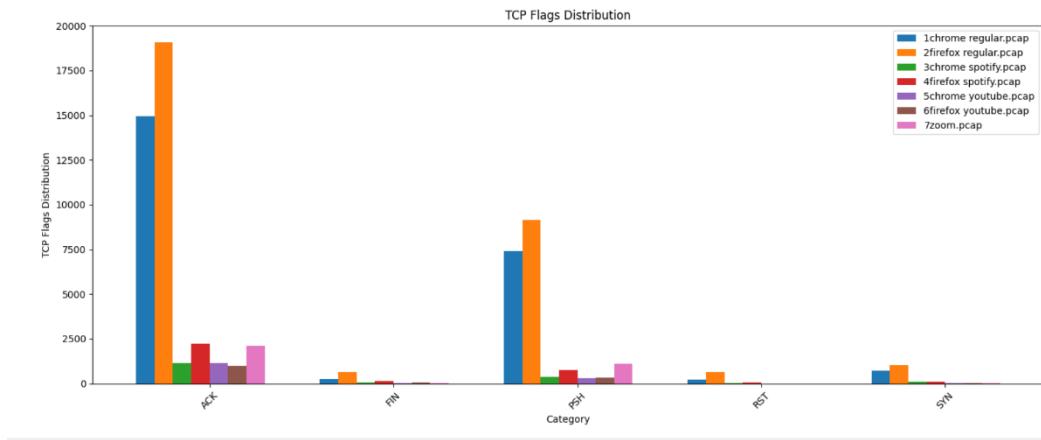
- Zoom מציג IAT ממוצע נמוך ביותר - מה שמצויב על שליחת רציפה ומהירה של חבילות קטנות.
- Zoom משתמש ב- UDP ובפרוטוקול RTP (Real-time Transport Protocol) שמטרתו להקטין השהיונות ולהבטיח שהשידור בזמן אמת לא יושפע מתקלות רשת.
- הגלישה הרגילה ב-Chrome וב-Firefox גם היא מציגה IAT נמוך יחסית, מכיוון שהדפדףים שולחים בתדרות גבוהה עבור בקשות HTTP.

### :TCP Flags Distribution. 3

הגרף מציג את התפלגות דגלי ה-TCP (TCP Flags) עבור סוג תעבורת שונים שנלכדו בקובצי PCAP.

dagli TCP הם סיביות (flags) בគורת של חיבור TCP, אשר מציינים את מצב החיבור והתנהגות הזרימה של התעborah בין לקוח לשרת.

מ ניתוח הגרף ניתן להסיק מסקנות לגבי אופי הגלישה, שימוש בפרוטוקולים שונים, ודפוסי התעborah של כל יישום.



#### מסקנות מהגרף:

##### 1. ישנה כמות גדולה של ACK לעומת PSH בכל סוג הגלישות.

- ניתן לראות בגרף כי מספר החבילות עם דגל ACK (Acknowledgment) על כל חיבור שנשלחת, גבוהה משמעותית בהשוואה ל- PSH.
- הסיבה לכך היא שמנגנון TCP- מחיבב אישור (ACK) על כל חיבור שנשלחת, ולכן דגל ACK קיים כמעט בכל החבילות.
- לעומת זאת PSH, משמש רק כאשר היישום רוצה "לדחוף" נתונים מיד למעלה ברמת האפליקציה, ולכן הוא פחות שכיח.

##### 2. בגלישה הדפדניים הרגילה יש יותר SYN מאשר בשאר הגלישות.

- SYN הוא דגל שנמצא רק בחבילות הראשונות של חיבור TCP חדש כחלק מתהליך לחיצת היד המשולשת.
- בגלישה בדף יש ביקורים תכופים באתרים רבים, וכך שכל אתר חדש מחיבב יצירת חיבור TCP חדש, ולכן יש יותר חבילות SYN.
- לעומת זאת, בשירותי סטרימינג כמו YouTube, Spotify, Zoom - החיבור נשמר לאורך זמן ולכן כמות ה SYN- נמוכה יותר.

### 3. כמות קטנה של RST, FIN ו-NYS באופן כללי.

- RST מופיע בעיקר כאשר חיבור נסגר באופן חריג, והוא פחות שכיח.
- FIN מופיע צד אחד סגור את החיבור בצורה מסודרת, אך בתעבורת אינטרנט מודרנית, דפדףים ויישומים נוטים לשמר חיבורם פתוחים כל עוד ניתן, וכך יש מעט חבילות עם דגל זה.
- NYS מופיע רק בתחילת חיבור TCP וכן הכמות שלו נמוכה יחסית לכל הabiliaות.

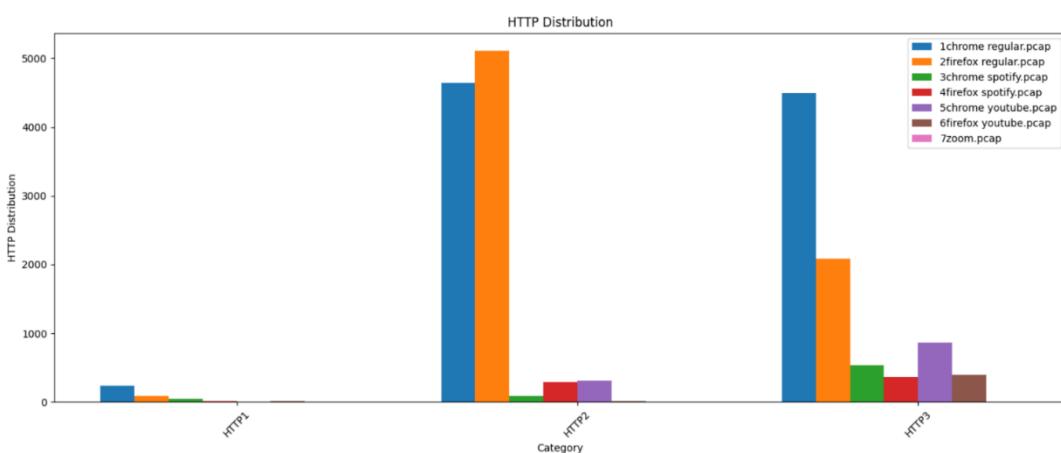
### 4. בגלישה דפדף מכל סוג, יש יותר מכל סוג ה-Flags מאשר בשאר סוג הגלישות.

- בגלישה בדפדף יש הרבה מאוד בקשות S/HTTP שמתבצעות דרך TCP וכן יש יותר שימוש בכל סוג הדגלים בהשוויה לשירותים מסוימים סטרימינג (שבהם יש פחות חיבורים חדשים).
  - גלישה דפדף יוצרת המון חיבורים קצרים לאטרים שונים, לעומת סטרימינג שבו יש חיבור ממושך אחד, מה שסביר את השכיחות הגבוהה של FIN, PSH, ACK, SYN ו-ACK.
5. גלישהspotify youtube Zoom , ואינם דומיננטיים בגין זה עקב העדפתם לפרטוקול להברת המידע.

### : HTTP Distribution.4

הגרף מציג את התפלגות פרוטוקולי HTTP (HTTP1, HTTP2, HTTP3/QUIC) עבור סוג תעבורת שונים שנלכדו בקובץ PCAP.

פרוטוקולים אלה מייצגים את הדרך שבה הדפכנים והאפליקציות מתקשרים עם השירותים ומספקים מידע על ההעדרה של כל דפדף ויישום לפרטוקול מסוים.



## **מסקנות מהגרף:**

**1. לכרכום יש העדפה לשימוש בחבילות (QUIC) HTTP3 ולפיירפוקס יש העדפה ל-HTTP2 (TLS)**

- Google Chrome תומך כברירת מחדל ב-QUIC (HTTP3) שהוא פרוטוקול מבוסס UDP שתוכנן במיוחד כדי לשפר את ביצועי הגלישה.
- Firefox לעומת זאת, נותן עדיפות ל-HTTP2 (TLS).
- הgraf מציג בבירור של Chrome יש יותר חבילות HTTP3 מאשר Firefox, ואילו Firefox משתמש יותר בHTTP2.

---

**2. לגלישות דףדף יש כמהות רבה יותר באופן מובהק מאשר לשאר הגלישות עקב כך שישנו ביקור בכמות אටרים לעומת אתר אחד.**

- גלישות דףדף מייצרת הרבה חיבוריו HTTP שונים, לאחר והמשתמש עובר בין מספר אתרים שונים.
- בשירותים כמו Zoom, YouTube, Spotify ו-Zoom יש חיבור ממושך אחד, ולכן יש פחות בקשות HTTP יחסית.
- הgraf אכן מראה כמהות גדולה משמעותית של חבילות HTTP בגלישה רגילה בהשוואה לשאר היישומים.

---

**3. כאשר פיירפוקס משתמש ביוטיוב, כמעט ואינו משתמש בחבילות HTTP2 למרות העדפה הבורורה שלו ל-TLS-ודעת מכיוון ש�וטיוב משתמש בשרתוי גול להעברת המידע.**

- YouTube פועל על שרתוי Google אשר מעדיפים (HTTP3) QUIC (HTTP3) מאשר פיירפוקס.
- Firefox נאלץ להשתמש ב프וטוקול המועדף על YouTube (HTTP3), גם אם הוא עצמו מעדיף HTTP2.
- הנתונים בgraf תומכים בכך: כמעט מבודדת חבילות HTTP2 נמצאות בתעבורת YouTube בפיירפוקס.

**4. Zoom בעקבות הפרוטוקול שלו DTLS-SRTP לא מאפשר לפענח את ההקלטות.**

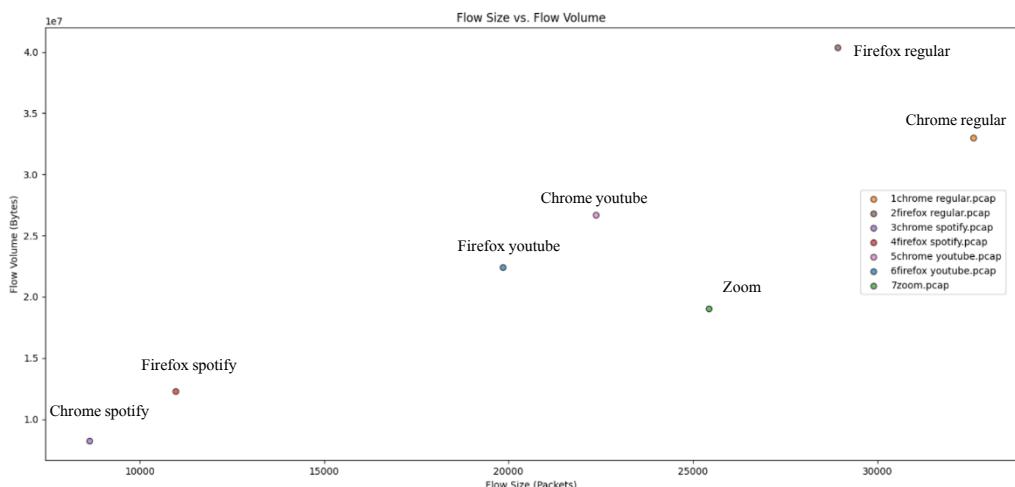
- Zoom מנצח את התעבורת שלו באמצעות DTLS-SRTP ולא משמש(HTTP2) בלבד להעברת המדיה.

- כתוצאה מכך, כמעט ואין חבילות HTTP בתעבורה, Zoom אחר וכל הנתונים המועברים מוצפנים ואינם מזוהים כHTTP רגיל.
- הגרף אכן מציג נוכחות מינימלית של חבילות HTTP בתעבורה zoom.

#### :Flow size vs Flow volume.5

הגרף מציג את הקשר בין מספר החבילות (Flow Size) לבין נפח הנתונים (Flow Volume) לבן נפח תעבורה (Flow Volume) לכל סוג תעבורה

- **ציר - X (Flow Size) גודל הזרימה:** מסמל את מספר החבילות שנשלחו בתעבורה.
- **ציר - Y נפח הזרימה :** (מצין את כמות הנתונים (בבייטים) שהועברה במהלך הזרימה).
- **הנקודות בצבעים שונים מייצגות את סוג התעבורה השונים, בהתאם למקרא.**



**מסקנה מהגרף:**

לכל סוג תעבורה יש פלח מסוילו בגרף:

הבדלים המשמעותיים בין סוגי התעבורה:

**1. YouTube** צורך נפח נתונים גדול אך עם **כמות חבילות בינונית**, לשאר, מכיוון שהוא מושדר **VIDEO + אודיו**, כאשר הוידאו דורש הרבה יותר משאבים. הוא משתמש ב-*Adaptive Streaming* (DASH), שמחולק את הסרטון **למקטעים גדולים** ולכון כמות החבילות היא בינונית – לא קטנה כמו ב-Spotify – ולא עצומה כמו ב-Zoom.

**2. Spotify** מייצר תעבורה נפח נתונים נמוך משל יוטיוב עם פחות חבילות, מכיוון שהעברת המוזיקה נעשית בצורה של **Streaming** רציף, שבו נשלחות אודיו, ללא צורך בהעברת וידאוCBD או מידע ויזואלי להבדיל מ-YouTube או Zoom.

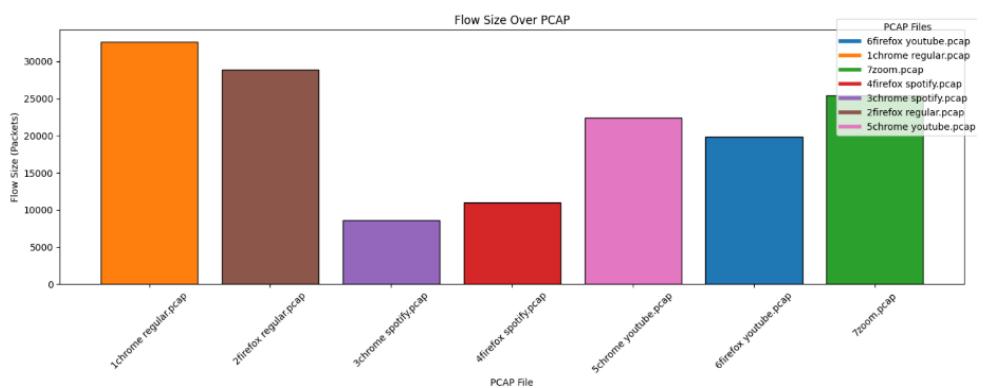
**3. Zoom** המבוסס על UDP שולח הרבה הרבה **חבילות קטנות** ביחס לשאר על מנת לשמור על תקשורת רציפה, וכך להעביר אודיו וידאו בתווך התקשרות, ולכון יש לו Flow Size גובה אר Flow Volume נמוך.

## **: Flow Size Over PCAP.6**

הגרף מציג את **כמויות החבילות (Flow Size)** שנשלחו עבור כל קובץ PCAP שנאסף במהלך הניתוח.

- **ציר-X** קובצי PCAP השונים, כל אחד מייצג סוג תעבורת אחר.
- **ציר-Y** מספר החבילות (packets) שנשלחו במהלך כל זרימה.

בשילוב זה ניתן **בשלוש האפליקציות Zoom, YouTube ו-Spotify**, תוך כדי התעלמות מgilisha רגילה עקב אופי התעבורת המשתנה שלה.



1. ל Zoom יש את כמויות החבילות הגדולה ביותר, ל YouTube-camot binoniyot ו-Spotify-cmotot kntna

### **Zoom שולח את מספר החבילות הגבוה ביותר**

- Zoom משתמש ב프וטוקול UDP ואיתו חבילות קטנות מאוד נשלחות בתדירות גבוהה כדי להבטיח תקשורת בזמן אמת.
- לכן, כמויות החבילות הכוללות (Flow Size) היא הגבוהה ביותר מבין כל השירותים.

### **YouTube נמצא באמצעות כמויות חבילות בינוניות**

- YouTube מבוסס על QUIC או HTTP2/HTTP3 כלומר משתמש ב *-buffering* חכם ובזרימה של חבילות בינוניות-גדולות.
- הסרטונים מועברים בקצבות של חבילות גדולות ולכן הוא צריך פחות חבילות יחסית ל Zoom אך יותר מ Spotify

### **Spotify שולח הכי מעט חבילות מבין השלשה**

Spotify משתמש באסטרטגיית *buffering* ייעילה, בה הוא מוריד חלקים גדולים של השיר מראש, מה שפחית את כמות החבילות הנשלחות בזמן אמיתי.

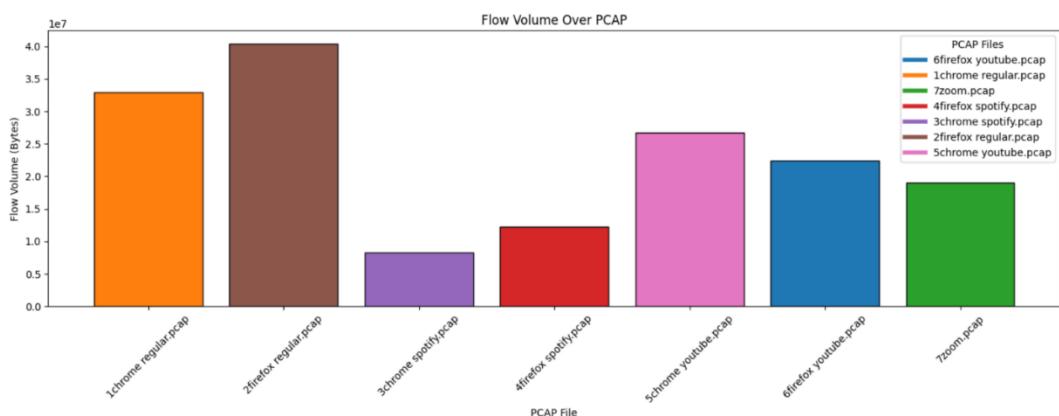
לכן, הוא צריך את כמות החבילות הקטנה ביותר בין שלושת השירותים.

### **: Flow Volume Over PCAP. 7**

הגרף מציג את נפח התעבורה (Flow Volume) שנשלח עבור כל קובץ PCAP במהלך ההקלטות.

- **ציר - X** קובצי PCAP השונים, שככל אחד מהם מייצג סוג תעבורה.
- **ציר - Y** נפח התעבורה הכללי (בבייטים), ככלומר כמה נתונים הועברו במהלך השימוש בכל אפליקציה.

בשילוב זה נتمكن בשלוש האפליקציות Spotify, YouTube ו Zoom כדי הצלמות מגילשה רגילה עקב אופי התעבורה המשתנה שלה.



YouTube צריך נ陪同ים גדול יותר מאשר Spotify,ici קטן, Zoom ביןוני.

#### **1. YouTube צריך את נ陪同ים הגודל ביותר**

YouTube משדר וידאו + אודיו, ולכן נפח הנתונים גבוה בהרבה מאשר סטרימינג של אודיו בלבד.

הוא יידאו מועבר באיכות גבוהה, תוך שימוש ב프וטוקול QUIC או DASH או שגורם לטעינה של מקטעים גדולים בבת אחת.

לכן, נפח הנתונים של YouTube הוא הגודל ביותר בין שלושת השירותים.

#### **2. Zoom נמצא באמצע עם נ陪同ים ביןוני**

- Zoom מעביר שיחות וידאו ואודיו, אך עשויה זאת באמצעות חבילות קטנות ב프וטוקול UDP
  - נפח הנתונים אינם עצום כמו ב-YouTube- Spotify. אך עדין גבוהה יותר מאשר משום שהשירות משדר באופן רציף ולא דחיסה אגרסיבית.
  - לכן, נפח התעבורה של Zoom בינוני – יותר מאשר Spotify- אך פחות מאשר YouTube.
  - צורך את נפח הנתונים הקטן ביותר Spotify משדר רק אודיו.
- .3. Spotify

#### **: IP Protocols Distribution.8**

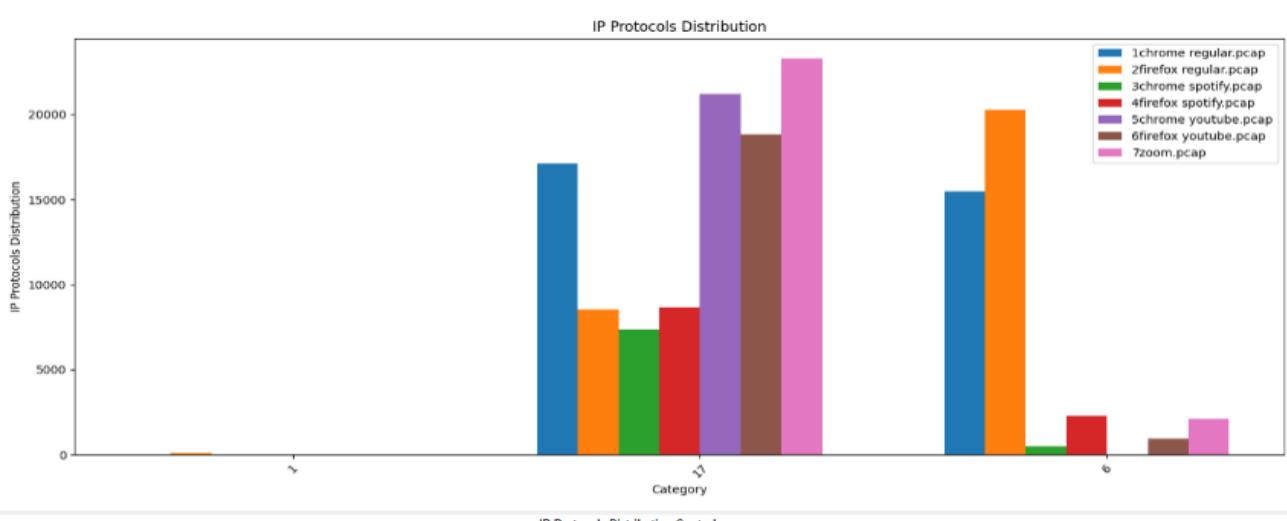
**UDP זה 17**

**TCP זה 6**

**ICMP זה 1**

הגרף מציג את התפלגות ה프וטוקולים ברמת IP כולם כמה מהחבילות בכל קובץ PCAP השתמשו בכל פרוטוקול

- (TCP) Transmission Control Protocol – המבצע אמין בקרת זרימה, המשמש בעיקר לHTTP(S).
- (UDP) User Datagram Protocol – פרוטוקול מהיר אך לא מהימן, נפוץ לשידורי וידאו, אודיו וגיימינג.
- (ICMP) Internet Control Message Protocol, המשמש בעיקר לשילוח הודעות בקרת רשת כמו Ping.
- הצבעים בגרף מייצגים את סוג התעבורה השונים.



## **מסקנות מהגרף:**

### Mozilla Firefox - מושפע מ-UDP ו-TCP יותר

- Chrome אכן שולח יותר תעבורת UDP בהשוואה ל-Firefox, מה שמייד על כך שהוא משתמש ב프וטוקול QUIC (HTTP3) שמצוין על UDP.
- Firefox לעומת זאת, משתמש בעיקר ב-TCP-מה שמרמז על העדפותו ל-HTTP2, שעובד על TCP ועל TLS.

---

### Microsoft Zoom ו-Youtube משתמשים ברוב מוחץ של udp לעומת tcp

- Zoom מבօיס UDP כי הוא משתמש בפרוטוקול DTLS-SRTP (Datagram Transport Layer Security - Secure Real-time Transport Protocol), המותאם לתקשורת אודיו/וידאו בזמן אמת.
- YouTube משתמש בעיקר ב-UDP-בגלל QUIC (HTTP3) שמאפשר טעינה מהירה יותר והפחיתה השהיות (Latency) בהזרמת וידאו.
- כמוות ה-TCP-קטנה יחסית, לאחר רוב התעבורת מתבצעת **בהזרמת וידאו**.

### **משמעות UDP**

### Spotify משתמש ברוב מובהק של UDP לעומת TCP

- Spotify משתמש ב-UDP כדי לאפשר שידור אודיו מהיר ויציב, לאחר שהאפליקציה יכולה להתמודד עם אובדן קל של חבילות מוביל לפגוע בחווית המשתמש.
- Spotify משתמש כמעט תמיד ב-TCP, אך עיקר ההעברה מתבצעת באמצעות פרוטוקולי UDP להזרמה מהירה עם השהייה מינימלית.

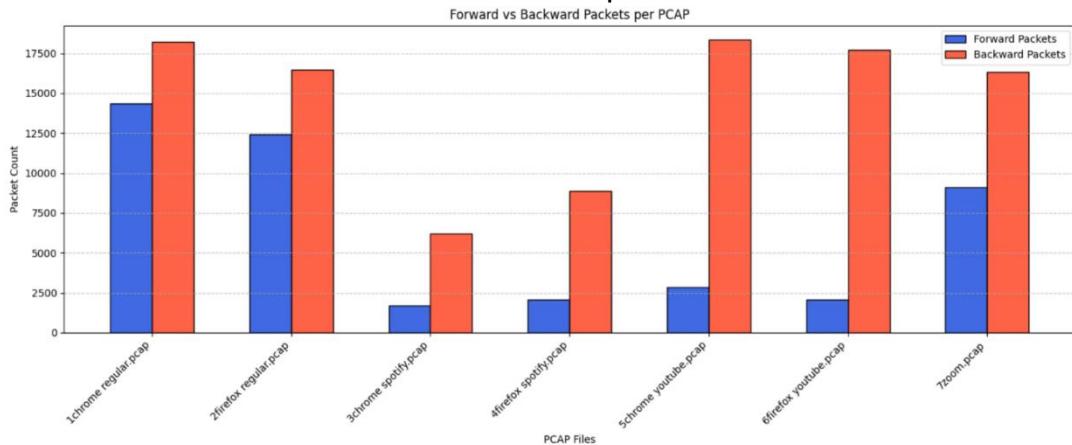
### בגלישה רגילה יש רוב מוחלט של TCP

כפי הגדפנים מבססים את התקשרות שלהם על פרוטוקול HTTP/HTTPS אשר דורשים אמינות גבוהה, אישור קבלת חבילות (Acknowledgment) מנגןוני תיקון שגיאות וברכת זרימה, המוצבטים על ידי פרוטוקול TCP.

## : Forward vs Backward Packets per PCAP .9

הגרף מציג את היחס בין חבילות שנשלחו מהלך (Forward Packets) לבין חבילות שהתקבלו מהשרת (Backward Packets) עבור כל תעבורת

- **חול –** (חבילות שנשלחו מהלך (המחשב/דף)).
- **כתום –** (חබילות שנשלחו אל הלך).
- ההשוואה בין הערכיהם מסייעת להבין את אופי התקשורת של כל שירות ואת מידת עומס הנתונים שנשלח מ/אל הלך.



### **מסקנות מהגרף:**

בתעבורת דף דפן יחס הזרימה קרוב ל-1, ככלMORE שליחת וקבלת חבילות נעשית בצורה מאוד מאוזנת

- בגלישה רגילה, (Chrome/Firefox Regular) הלך שולח בקשות HTTP רבות, והשרת משיב בחבילות נתונים המכילות 많은 נתונים.
- תעבורת דף דפן נוטה להיות מאוזנת יותר מכיוון שהיא מערבבת בקשות רבות וקובלת נתונים במקטעים קטנים יחסית.

בՅוטיוב יחס השליחה קרוב מאוד ל-0, ככלMORE הלך שולח מעט חבילות אך מקבל כמות עצומה של חבילות מהשרת.

- YouTube מבוסס על סטרימינג וידאו, שבו הלך שולח מעט בקשות למשל בקשה GET לסרטון, אך השרת משיב בכמות עצומה של חבילות המכילות את הוידאו עצמו.
- פרוטוקולים כמו (HTTP3) DASH Streaming | QUIC גורמים לכך שהשרת שולח נפח נתונים גבוה בצורה רציפה, בעוד שהלך שולח כמעט ולא שולח חבילות.
- לכן, יחס ה Forward-Backward הוא מאוד נמוך, כי רוב התעבורת היא בכיון אחד – אל הלך.

בגילישות של Spotify יחס השליחה קרוב ל-0, אך לא כמו ביוטיוב – כלומר הלקוח שולח מעט חבילות ומקבל הרבה, אך פחות מאשר ביוטיוב.

- Spotify עובד במנגןון, buffering בו השיר יורד במקטעים גדולים במקום להזרים נתונים באופן מתמשך כמו YouTube
- הלקוח שולח מעט חבילות, אך השרת מהזיר כמות גבוהה של חבילות אודיו.
- בנגד לביוטיוב, בוידאו צריך להמשיך לקבל חבילות כל הזמן, בסופטיפי יש פרקי זמן של השמעה ללא צורך בנtones נוספים, ולכן מספר החבילות החזרות נמוך יותר.
- לכן, יחס השליחה נמוך, אך עדין גבוהה מעט יותר מאשר ביוטיוב.

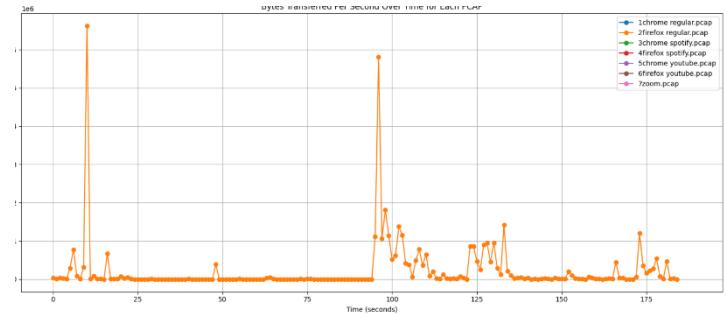
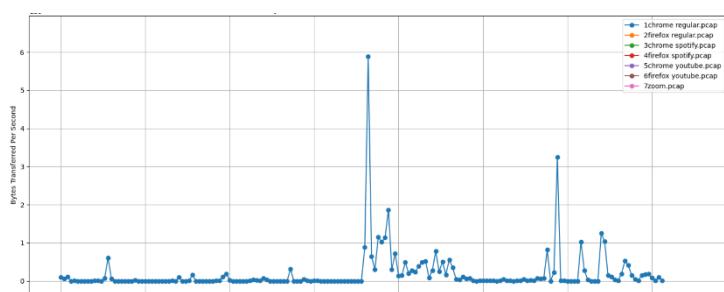
לגביו, Zoom קשה לחלץ דפוס קבוע וחד-משמעותי של יחס Forward/Backward, כיוון שיש לו זה מושפע ממספר משתנים הקשורים לאיכות השיחה והתנאים ברשות:

- כמות החבילות הכוללת שנשלחת בZoom-תלויה בשירות באיכות החיבור של המשתתפים, ברוחב הפס הזמן, באיכות הוידאו והאודיו (למשל, מצלמה, מיקרופון), וביכולת הדחיסה והקידוד בזמן אמת.
- עם זאת, ניתן לבדוק אם Zoom בכלל שואף לתקשרות סימטרית יותר (יחס Forward/Backward קרוב ל-1) ביחס לשירותי סטרימינג חד-כיווניים כמו YouTube-Spotify. Zoom הוא שירות אינטראקטיבי, שבו כל המשתתפים שולחים ומקבלים נתונים באופן רציף במהלך השיחה.

### :Flow Volume Per Second.10

גרפים אלו מציגים את כמות הנתונים (בבייטים) שהועברה בכל שנייה לאורך זמן עבר כל תקופה. המטרה היא לזהות את מאפייני הדפוס של תקופה שונה – האם היא איחודה, מתרצת, יציבה או משתנה בתדרות גבוהה.

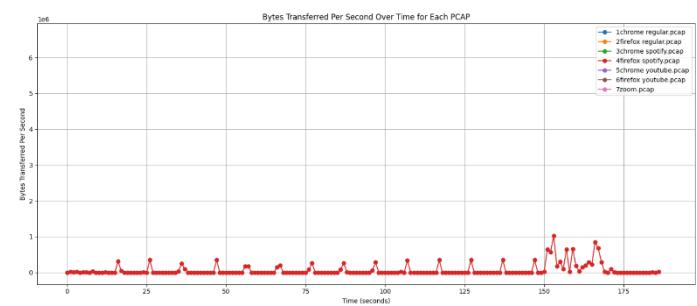
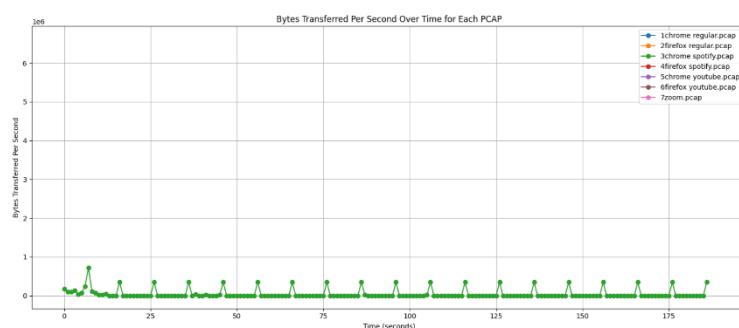
### :Flow Volume Per Second.10 גישה רגילה:



**מסקנה:** תעבורת הגלישה מציגה קפיצות לא סדירות בכמות הבטים המועברת, עם "שיאים" ברורים ופרק זמן של כמעט אפס פעילות.

- סיבה:** בגלישה רגילה, הנתונים נשלחים ונטענים באופן גושי (burst) בעיקר כאשר משתמש נכנס לעמודים חדשים או טווען תוכן. אין רצף של נתונים לאורך כל האינטראקציה כמו בשאר הגלישות בהשוואה שלנו.

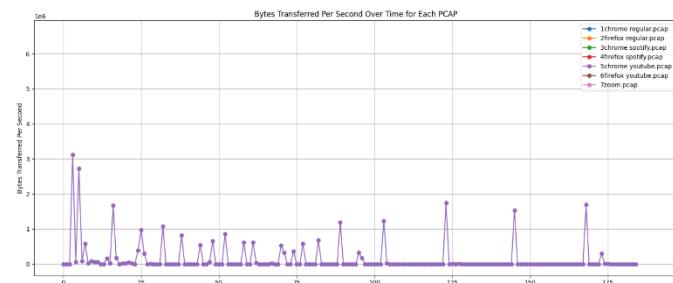
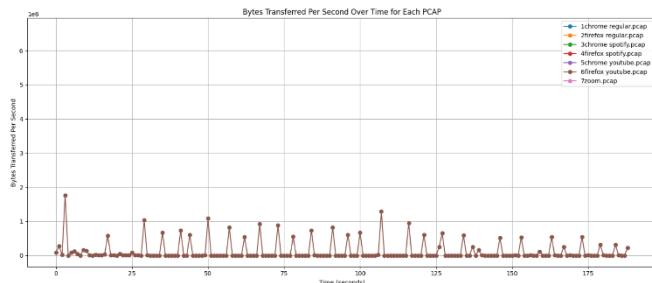
## ♫ סופיטיפי: (Chrome + Firefox)



- מסקנה:** נפח התעבורה של סופיטיפי מציג תבנית מחזורית – שיאים נמכרים, קבועים, חוזרים כל מספר שנים, בעלי מרוחקים קבועים יחסית ביניהם כאשר ברקע התעבורה יציבה מאוד.

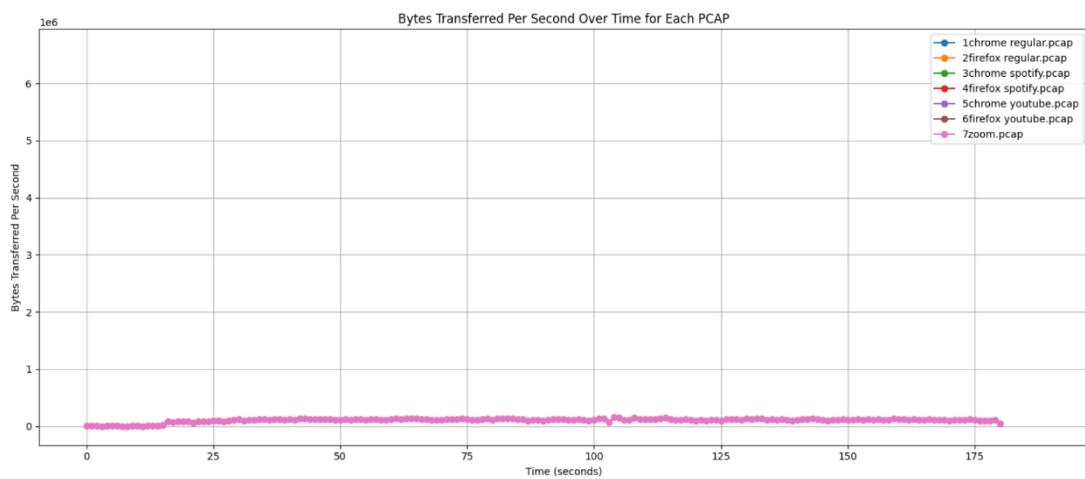
- סיבה:** סופיטיפי משתמש במנגןון של, "buffering"(Clomar טווען בלוקים קטנים של מזיקה מראש באופן קבוע, מה שיוצר תעבורה יציבה יחסית. המחזוריות משקפת את אופן טיענת קטעי השירים, ולכן קצב המחזוריות קבוע פחות או יותר בהתאם לאורך המקטעים הנטענים. ומכיון שגם מזיקה יש לה BPM קבוע המתבטא בתעborות הנתונים וברציפותה ואחדותה באופן ציורי.

## 📺 יוטיוב: (Chrome + Firefox)



- **מסקנה:** תעבורת יוטיוב מאופיינית בשיאים גבוהים משל ספוטיפי, בעלי גובה לא קבוע, עם מרוחקים לא קבועים, עם פעילות רבה בהתחלה ולאחר מכן ירידת משמעותית בפעולות.
- **סיבה:** יוטיוב מבצע – "pre-buffering" הוא טובע מיקטעים גדולים של הוידאו מראש, ואז מפסיק לזמן מה, עד שיש צורך להוריד עוד חלק. זה גורם לתעבורה להיות קפיצית אך חזקה. בנוסף עקב כך שהוא טובע יידאו, אודיו מולו הוא זניח בנפח, ולכן השיאים הגבוהים שלו לעומת זום וספוטיפי מאד ברורים. בנוסף כיוון שהאודיו זניח והתמונה היא דבר משתנה כאשר מדובר בסרטון, הגיוני מאד שהמרוחקים והשיאים יהיו לא קבועים גם כן.

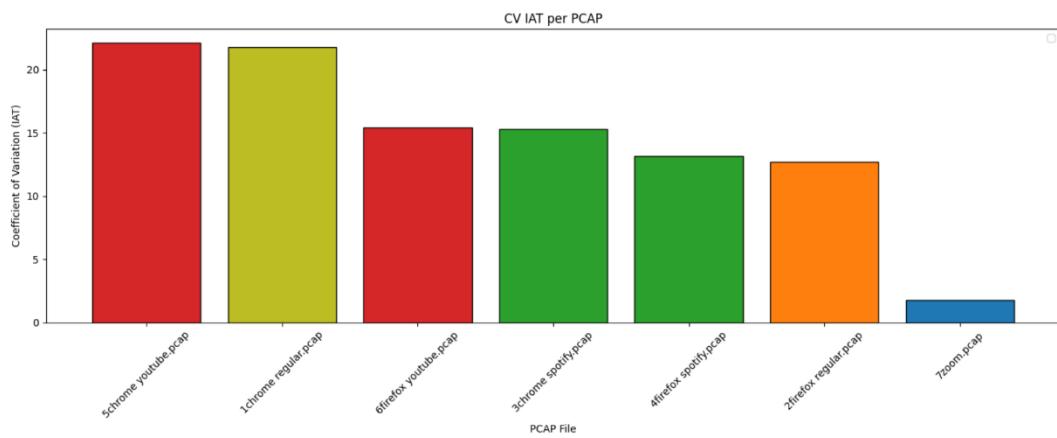
**זום:**



- **מסקנה:** תעבורת זום היא בעלת קצב קבוע יחסית, כמעט ללא תנודות.
- **סיבה:** כיוון שמדובר בשיחת וידאו בזמן אמת (real-time) זום חייב לשומר על קצב תעבורה אחיד כדי להבטיח חווית שיחה רציפה. לכן, הוא משדר באופן עקבי ובקצבים קבועים, גם אם כמות הנתונים המועברת בשנייה יחסית נמוכה לעומת יוטיוב למשל, בנוסף הוא משתמש ב프וטוקול udp ומתעדף העברת נתונים רציפה לעומת אינטראקטיבית, לכן ניתן להבין את כמות החבילות הקטנות הרבות שעוברות לעומת האלטרנטיבה הנאייבית כמוות חבילות קטנה עם נפח גדול.

## **:(Burstiness Factor)IAT-CV.11**

כמו שצוין במאמרים CV בהקשר של ניתוח רשות מחושב ע"י חלוקת השונות של הזמןם בין החבילות בממוצע של הזמןם בין החבילות.  
**מהות הגרף** היא להציג את מידת התפרצויות (Burstiness) של זמן ההגעה בין החבילות (Inter-Arrival Times) בקובצי PCAP השונים, כפי שנמדד באמצעות מקדם השונות (CV). ככל שהערך גבוה יותר, כך שונות הזמן ההגעה בין החבילות גדולה יותר, מה שמעיד על דפוסי שליחה פחות סדריים.

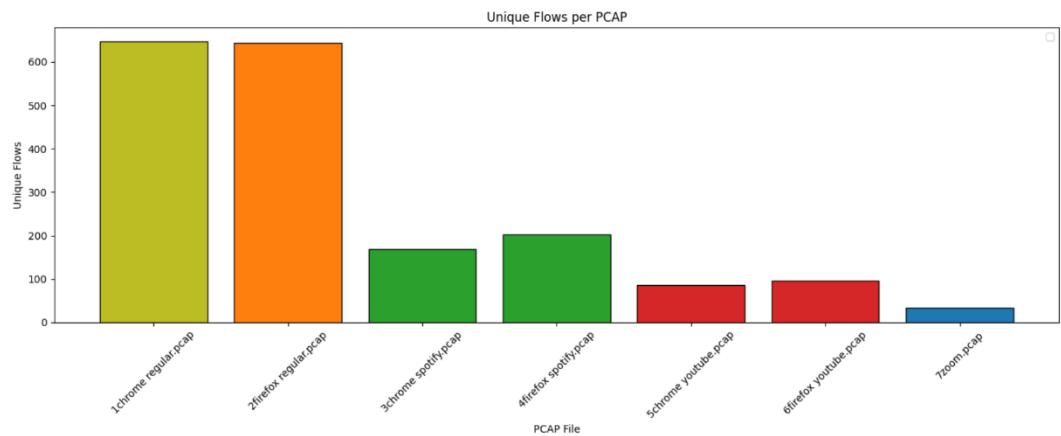


### **מסקנות מהגרף:**

Zoom מקיים תבעורה הרבה יותר רציפה, יציבה וחדידה (חסרת התפרצויות), זאת כדי לאפשר תקשורת רציפה וברורה בין משתמשי השיחה.

## :Unique Flows.12

בחרנו בגרף זה ליצג את כמות הזרימות הייחודיות לפי ( Src ip, Ds tip, src port, dst port ), לכל סוג תעבורת.



### **מסקנות מהגרף:**

לכל סוג תעborah, כמות הזרימות הייחודיות שבה מעניקה לה טביעה אצבע ייחודית משלה, שכן ניתן להבדיל בין כל הгалויות בגרף זה ב יתר קלות.

## חלק 4 (נקודות מבט של תוקף):

בחלק זה נתבקשנו להקליט שוב תעבורה ולבדוק עד לאיזו רמה נוכל לגלוות באילו אפליקציות השתמשו הנטקף, כאשר התעבורת מוצפנת, בשתי מקרים:

כאשר יש לנו גישה בכלל pcap ל:

- גדים, זמני ההגעה וhash של tip, src port, dst port(ip,ds src) של כל חבילה (Flow ID).
- גדים וזמנים של כל חבילה.

כדי לעשות זאת השתמשנו במאגר הקלטות של אחד הסטודנטים, בו קיימות 25 הקלטות לכל סוג תעבורת מהబאים: (Chrome,edge,spotify,youtube,zoom), סה"כ 125. ישנו אלגוריתם למידת מכונה בהשראת המאמרים, בעזרת Random Forest Classifier על הקלטות.

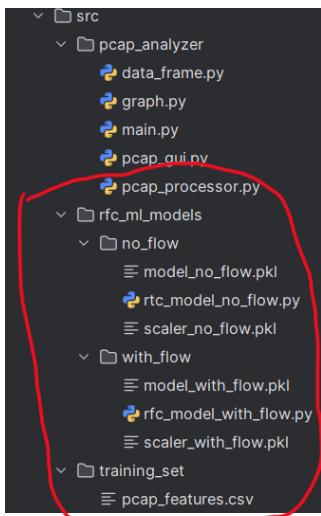
על מנת לאמן את האלגוריתם לקחנו 15 מכל סוג הקלטה (75 סה"כ), ביצענו חלוקה של 80% לאימון 20% לבדיקה (12 מכל סוג לאימון מול 3 לבדיקה), נגדיר קבוצה זו כ **Data Set** (בה ישן 75 הקלטות).

בעזרת ניסוי ותיהיה רבים של מאפייני התעבורת בהם נתקלנו שימושיים לאורך המטלה, הגענו לאלגוריתם שמצוה את התעבורת גם בקבוצת הבדיקה (3 מכל סוג) וגם ב50 שלא השתמשנו באחוזים גבוהים מדי.

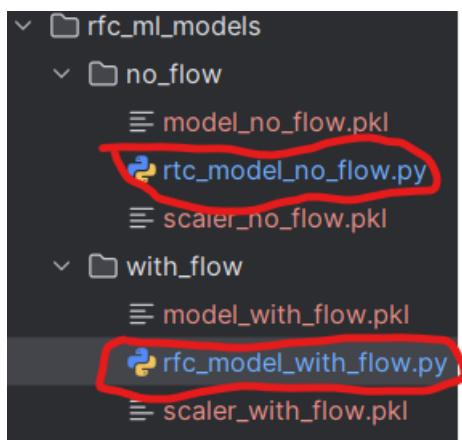
לפני הסבר המימוש נציין ש כדי להגיע לאלגוריתם למידת מכונה שמצילח לזהות תעבורת באופן כלל מערכתי/ כלל גיאוגרפי היינו זקוקים לגישה למשאבים רבים ומגווני מידע של הקלטות בעלי מאות אלפי הקלטות מאזורים גיאוגרפיים שונים, וממערכות שונות, כפי שלמדנו מהמאמרים. אך עבור מידדי המטלה הגענו לתוצאות לא רעות, ואת המסקנות שלנו נפרט לאחר הסבר המימוש.

## מימוש

בחלקו הראשון של הקוד, היה علينا למשתמש בניתוח גרפי ויזואלי על הקלטות המיצגות כל סוג תעבורת, אשר הציגו וניתנו בחלק 3, כתע נתעסק בקבצים המסומנים בעיגול.



בחלק זה של המטלה על מנת להישאר צמודים להנחיות, בשתי קבצי ה `rfc_model` המסומנים בתמונה, הינו קובץ `csv`(שתי) היקלט של האלגוריתמים למידה, הינו קובץ `pcap_processor`, אשר הוכיח מוקודם), אשר הופק ע"י ה `rfc_model`, אשר הוכיח עצמו שימושו בשתי חלק הנטלה (הניתוח הגרפי והחלק של התוקף).



עבור כל אחד מקובצי ה `pcap` שבקובוצת האימון והבדיקה (סהכ 75 קבצי `pcap` אשר ניתן להורדה בקישור בדף הראשון של PDF), ה `processor` מחלץ אך וرك את המידע אשר מותר לתוכף לדעת, לטור `training_features.csv` המשמש בתורו ה `Data Set` שלנו.

לכל אחד משתי המקרים בחרנו סט של מאפיינים על פיו הוא ימד את הדפוסים הקיימים המידע שבקובצי ה `pcap`.  
ביצענו `finetuning` לסט המאפיינים ול `hyper parameters` של עומק העצים שמחליתם באלגוריתם, מצאנו את המאפיינים הבאים כאליה ששימשו את אחוזי הסיווג שלנו בrama המיטבית:

FlowID עם

FlowID בלי

```

FEATURE_SELECTION = [
    # Packet stats
    "std_ps",
    # Flow-based features
    "fFlow_count",
    # Newly added flow-based packet size stats
    "std_of_flow_pkt_size_std"
]
  
```

```

FEATURE_SELECTION = [
    "avg_iat",
    "std_ps",
    "avg_ps"
]
  
```

בחרנו במאפיינים אלה אחרי ניסויים רבים של ניסוי ותיהיה, אשר הביאו אותנו לתוצאות המיטביות מבחינת סיווג התעborות, ומטריצת בלבול המבחן בסיווג תלוי אופי השירות במקורה ראשון ומטריצה המבחן בסיווג מדויק במקורה השני ככל הניתן.  
את התוצאות נציג לאחר הסבר המאפיינים.

עבור כל מאפיין שהוא בשימוש, השתמשנו בMinMax Scale על מנת לנормל אותו ביחס לשאר החבילות באותו pcap, ולאחר מכן חצנו אותו לRFC.  
הנורמל נעשה בדרך הבאה:

$$X_{\text{scaled}} = \frac{X - X_{\min}}{X_{\max} - X_{\min}}$$

כאשר  $\min$  ו  $\max$  מתייחסים לערך המקסימלי והמינימלי של המאפיין באותו pcap.

<u>הסבר</u>	<u>מאפיין השוואתי</u>
סטיית התקן של גודל החבילות (Packet Size). מוגדר עד כמה גDALי החבילות משתנים בתוך זרם התעבורה.	<b>Std_ps</b>
מספר הזרמים (Flows) בתוך הקובץ הנמדד. זרם מוגדר לפי הגדרת המטלה.	<b>Flow_count</b>
סטיית התקן של סטיות התקן של גDALי החבילות בתוך כל זרם. מוגדר עד כמה פיזור גDALי החבילות משתנה בין זרים שונים.	<b>Std_of_flow_pkt_size_std</b>
הגודל הממוצע של החבילות, מספק מدد לגבי הגודל הטיפוסי של חבילות בפרק זמן נתון.	<b>Avg_ps</b>
מקדש השונות של גודל החבילות, ( השונות חלקית הממוצע של גDALי החבילות).	<b>Cv_ps</b>
זמן הממוצע בין הגעת חבילה אחת לשניה. ממדד לעומס ולפרק הזמן בין שליחת נתונים.	<b>Avg_iat</b>
סטיית התקן של זמני ההגעה (IAT). מוגדר עד כמה הפרשי זמני ההגעה של החבילות משתנים לאורך הזמן.	<b>Std_iat</b>
מקדש השונות של הזמנים בין החבילות, ( השונות חלקית הממוצע של הזמנים בין החבילות).	<b>Cv_iat</b>
מספר החבילות הנשלחות או מתקבלות <b>בשניה</b> . ממדד לשיעור התעבורה בפרק זמן נתון.	<b>Packets_per_second</b>

עבור כל scenario העז בעזרת המאפיינים הללו מקיים את החלטות מסווג כל קובץ עד לעומק מסוים, ולבסוף משקלל את כל האפשרויות לטובה ביותר.

לאחר ניסוי ותיהיה רבים וכוכoon המאפיינים לדוק הסיווג הגבואה ביותר אלה התוצאות אשר הניבו את התוצאה במיטבית בשתי המקרים (בעמוד הבא):

## תוצאות הבינוני:

FlowID עם

FlowID בלי

### תוצאות אימון האלגוריתם

```

 Best Hyperparameters Found:
- max_depth: None
- max_features: sqrt
- n_estimators: 50
 Model Accuracy: 93.33%
 Train Accuracy: 100.00%
 Test Accuracy: 93.33%

• Feature Importance Ranking (Sorted):
flow_count: 0.3517
std_of_flow_pkt_size_std: 0.2802
std_ps: 0.2262
cv_iat: 0.1419

Confusion Matrix:
      chrome   edge   spotify   youtube   zoom
chrome       3     0       0       0       0
edge         0     2       1       0       0
spotify      0     0       3       0       0
youtube      0     0       0       3       0
zoom         0     0       0       0       3

 Model and Scaler saved.

```

```

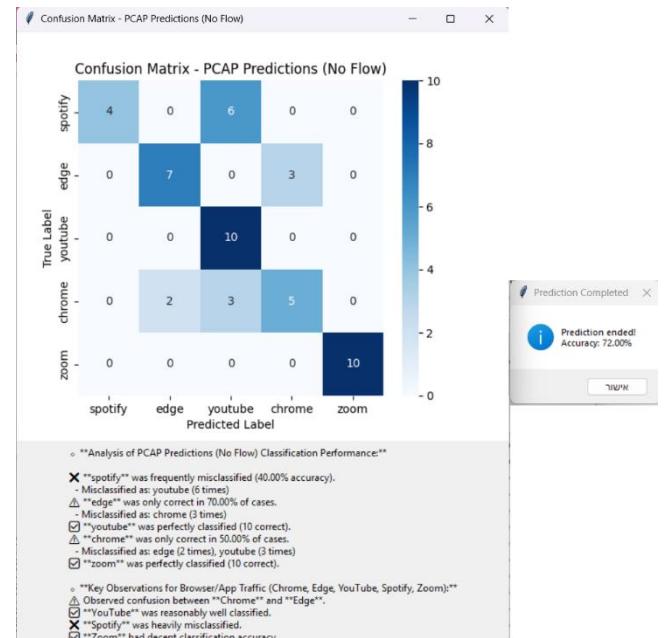
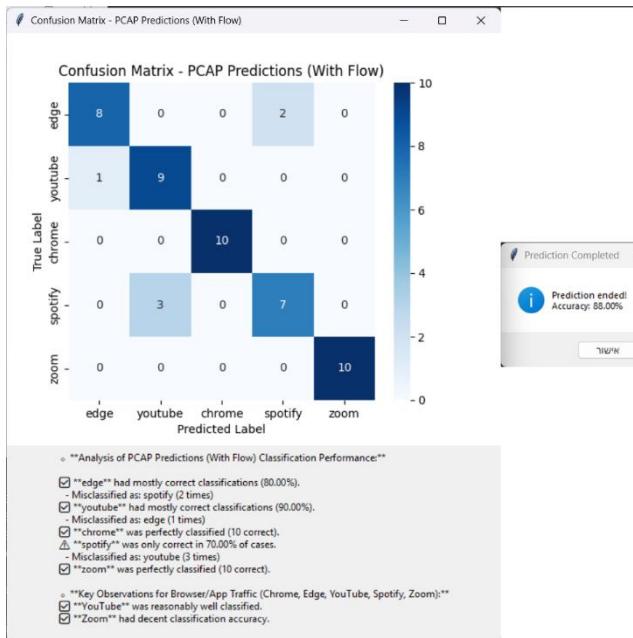
 Best Hyperparameters Found:
- max_depth: None
- max_features: sqrt
- n_estimators: 200
 Train Accuracy: 100.00%
 Test Accuracy: 73.33%

• Feature Importance Ranking:
std_ps: 0.3652
avg_ps: 0.3177
avg_iat: 0.3172

Confusion Matrix:
      chrome   edge   spotify   youtube   zoom
chrome       3     0       0       0       0
edge         2     1       0       0       0
spotify      0     0       3       0       0
youtube      0     0       0       2       1
zoom         0     0       1       0       2

```

וכאשר בחנו אותו בנוסף על 10 הקלות מכל סוג מהמארג, אשר לא הזנו לקוד חלק מה קיבלנו: Data Set



ניתן לראות בבירור של לאחר מציאת שילוב המאפיינים שמביא לאחוז דיקס איסויוג הגבואה ביותר בכל סוג מקרה:

- אחוז דיקס איסויוג גבואה משמעותית כאשר נתונים לנו ה Hash של ה ID Flow של כל חבילה.
- כאשר לא נתונים לנו ה ID Flow לכל חבילה, קשה לסייע את סוג האפליקציה הספציפית אך עדין ניתן לזיהות את אופי השירות.

לדוגמה בתמונה הימנית:

- ספוטיפי לא סוג נכון, ב 60 אחוז ממספר המופיעים שלו, אך עדין סוג השירות שמבצע Streaming .
- גם כאשר edge לא סוג נכון הוא סוג כדף (chrome) .

רוב טעויות איסויוג נעשו לאותו אופי שירות.

## מסקנות הכספי והמטרלה:

- גם כאשר התעבורה מוצפנת, לתקוף יש את הפוטנציאל לזרות את סוג האפליקציה/דף באחוזי סיווג מאד גבוהים. אם יש לו גישה למגר מידע בקנה מידה גלובלי של הקלטות, הוא יכול להגעה לאחוזי דיק בסיווג העולמים על 90%!
- כאשר לתקוף יש גישה אך ורק **לגדלי החבילות וזמן הגעה**, הוא יכול בעזרתו חישובים סטטיסטיים, ויזיה דפוסים, לזרות בהם אחוזי דיק מהמסקנה לעיל את אופי האפליקציה בה נעשה שימוש (גילישה אקראית/ צפיה בסרטון/ השמעת מזיקה/ שיחה רציפה ברשת). אך לזריהו סוג האפליקציה הוא זקוק לטביות אכבע יחודית הקשורה אליו באופן ישיר.
- כאשר לתקוף יש גישה **לגדלי החבילות, זמן הגעה ונתונים סטטיסטיים שטחיים** לגבי הזרימות (כלומר ללא מידע על כתובות השולח/ המקבל ועל התווך בו נעשות השליחות של המידע), הוא יכול לזרות אפילו את סוג האפליקציה באחוזי דיק מהמסקנה הראשונה.
- הנתונים הסטטיסטיים השטחיים העולים מניטוח הזרימות בתעבורת רשת, מייצגים טביות אכבע יחודית של שירותים שונים.
- גם כאשר התעבורה מוצפנת, לתקוף יש את הפוטנציאל לזרות את סוג האפליקציה/דף באחוזי סיווג מאד גבוהים. אם יש לו גישה למגר מידע בקנה מידה גלובלי של הקלטות, הוא יכול להגעה לאחוזי דיק בסיווג העולמים על 90%!
- כאשר לתקוף יש גישה אך ורק **לגדלי החבילות וזמן הגעה**, הוא יכול בעזרתו חישובים סטטיסטיים, ויזיה דפוסים, לזרות בהם אחוזי דיק מהמסקנה לעיל את אופי האפליקציה בה נעשה שימוש (גילישה אקראית/ צפיה בסרטון/ השמעת מזיקה/ שיחה רציפה ברשת). אך לזריהו סוג האפליקציה הוא זקוק לטביות אכבע יחודית הקשורה אליו באופן ישיר.
- כאשר לתקוף יש גישה **לגדלי החבילות, זמן הגעה ונתונים סטטיסטיים שטחיים** לגבי הזרימות (כלומר ללא מידע על כתובות השולח/ המקבל ועל התווך בו נעשות השליחות של המידע), הוא יכול לזרות אפילו את סוג האפליקציה באחוזי דיק מהמסקנה הראשונה.
- הנתונים הסטטיסטיים השטחיים העולים מניטוח הזרימות בתעבורת רשת, מייצגים טביות אכבע יחודית של שירותים שונים (אומם סיכמו בסוף המטרלה).

## כיצד נמנן את יכולותיו של התקוף לזהות את התעבורה?

### 1. Padding (הוספת "רעש" לתעבורה)

התקפה מתבססת על זיהוי דפוסי גודל חבילות זמני שלicha מדוייקים. אחת הדרכים למתן זאת היא Padding (הרחבת חבילות), שמטרתה:

- להפוך את החבילות לאחדות בגודלן או להוסיף גודל רנדומלי, וכן לטרנסס את מאפייני התעבורה הייחודיים לכל אפליקציה.

למשל, פרוטוקול Tor עושה שימוש ב, padding וכפי שראינו במאמר FlowPic, הדבר מקשה יותר על הסיווג.

### 2. טשטוש הזמן (Timing Ofuscation)

- מכיוון שדף הזמן (Inter-Arrival Times) של החבילות מהו הפעמטר לזיהוי האפליקציה, ניתן להוסיף עיכובים או לשנות חבילות מרוחקי זמן קבועים או במרקוזים אקראים, (Constant Bitrate),

אסטרטגיה זו מטשטשת את החתימה של אפליקציות כמו Zoom או Skype המאפשרות בתעבורה רציפה ו אחידה בזמן.

### 3. Multiplexing (איחוד זרמי נתונים)

- שימוש בשיטות כמו Multiplexing או חיבור מרבוה אפליקציות על גבי זרם תקשורת יחיד יכול להקשות על בידוד אפליקציות בודדות.

לדוגמה, שימוש ב VPN או CDN המשלבים תעבורה מאפליקציות שונות יכול להקשות על התקוף לזהות אפליקציה ספציפית מתוך זרם נתונים אחד גדול.

### 4. הצפנה מידע מזדהה (כמו SNI)

- פרוטוקולים כמו TLS 1.3 עם Encrypted ClientHello מסתירים מידע שבעבר היה גלוי וחושף לזיהוי, כגון ה-SNI (Server Name Indication).
- למרות ש ECH מבקשת על סיוג מוקדם, (Early Classification) עדין יותר מידע לא מוצפן ברמת ההандשאך (כמו שŁMDDנו במאמר Early Client Hello () לכך, שילוב ECH יחד עם Padding או טשטוש הזמן יכול להיות יעיל במיוחד).

### 5. יצירת "רעש" סינטטי (Dummy Packets)

- יצירת חבילות "דמה" שאין נשאות תוכן אמיתי אלא נועדו רק לבלב את הניתוח. חבילות אלו יכולות להיות בעלות מאפיינים דומים לחבילות אמיתיות, מה שיקשה על הזיהוי.
- שיטה זו נמצאת בשימוש בפלטפורמות כמו Tor כדי להוסיף אקריאיות לתעבורה.

### 6. שימוש בתעבורה סימטרית (Symmetric Traffic)

- הפיכת זרימת התעבורה ליותר סימטרית (שווה פחות או יותר בכמות הנתונים המועברת בשני הכוונים) יכולה להקשות על זיהוי על פי מאפייני תעבורה כמו כמות הנתונים לכיוון מסוים, המשמשת לזיהוי חד-כיווני של אפליקציות.

## 7. פיצול התעבורה

- פיצול חבילות נתונים גדולות למספר חבילות קטנות בגודל אחד יכול להקשות על זיהוי החתימה של אפליקציות שעושות שימוש בחבילות גדולות וקלות לזיהוי) כמו Netflix או Youtube.

## 8. שימוש בפרוקסি ובשכבות הצפנה רבות

- לדוגמה, שימוש בVPN עם TLS או שילוב Tor או VPN מעלה את רמת הקושי בסיווג על ידי טשטוש מאפייני התעבורה הבסיסיים.

## 9. שינוי דינמי של מאפייני זרימת הנתונים (Adaptive Traffic Shaping)

- שינוי אקטיבי של מאפייני התעבורה באופן דינמי יכול לסכל מודלים סטטיסטיים של זיהוי. לדוגמה, מעבר אקראי בין גודלים שונים של חבילות ותבניות שליחה, על פי התראות או סימנים ראשוניים של ניסיון זיהוי.

## טביעות אצבע ייחודיות לכל סוג אפליקציה מהמטלה:

### **דפדף בגלאי אקרואית:**

- תעבורת קצרה מועד ומשתנה בעלת פרצי מידע קצרים.
- מאופיינת בגדי חבילות משתנים מאוד (קטנות וגדלות לסייעון).
- זמני הגעה (Inter-Arrival Times) לא קבועים, בעלי שונות גבוהה.
- זרימות קצרות יחסית (הרבה זרימות חדשות נפתחות ונסגרות).
- יחס מאוזן יותר בין תעבורות שליחה לקבלה.
- כמות החבילות קטנה יחסית לכל Session אך עם מגוון רחב של גודלים.

### **(צפיה בסרטונים) YouTube**

- תעבורת עם חבילות גדולות וממושכות.
- זרימה רציפה לאורך זמן, עם מרוחקים קבועים יחסית בין החבילות.
- מאפיין התנагות של "Buffering" פרצי מידע גדולים בתחילת כל סרטון, ואז מעבר לתעבורה קבועה ואיטית יותר.
- **יחס אסימטרי מאוד** של שליחה לעומת קבלת נתונים.
- זמני הגעה קבועים יחסית לאורך כל הסרטון לאחר שלב Buffering.

### **(Audio streaming) Spotify**

- תעבורת בעלת חבילות בינוניות-גדולות (פחות מווידאו, אך גדולות יותר מגלישה רגילה).
- זרימה יציבה יחסית, קבועה וממושכת, עם התנאגות רציפה לאורך השמעה.
- זמני הגעה קבועים ואחדים יחסית, יציבות לאורך האזנה.
- **יחס אסימטרי ברור** בעיקר קבלת נתונים מטה שליחת נתונים.
- כמות החבילות לכל פרק זמן נתון אחידה למדי לאחר התחלת קרצה של טעינה.

### **(Video Conferencing) Zoom**

- תעבורת רציפה של חבילות קטנות-בינוניות בתדריות גבוהה.
- זמני הגעה קצרים וקבועים רוב הזמן מדווח בתקשות זמן-אמת ללא השהיה.
- **יחס תעבורת סימטרי** באופן ייחסי (כמות גבואה של שליחה וקבלת נתונים מקבילים).
- מאפייני החבילות יציבים בגודלן (קטנות יחסית לעומת סטרימינג וידאו כמו YouTube)
- זרימה דו-כיוונית יציבה לכל אורך השיחה, ללא הפסקות משמעותיות.