פרויקט גמר רשתות תקשורת

מגישים:
אביב נאמן - 318446804
נועה פטיטו - 322694530
גיל אהרון - 209012095
אמנון פוזאילוב - 322275504

קישור לקבצי ה pcap ולsslkeylogfile הרלוונטיים למטלה.

קישור לrepository עם קבצי הקוד בgithub.

תוכן עניינים:

תשובות לחלק 1:

1.גורמים להאטה בהעברה:

בקרת עומס בTCP-

TCP משתמש במנגנוני בקרת 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 המקבל : אם המקבל מפרסם חלון קטן בגלל מגבלות בזיכרון הbuffer שלו, זה יאט את ההעברה, גם אם השולח יכול לשלוח נתונים בקצב גבוה יותר.
- איבוד מנות ושידור חוזר: איבוד מנות גורר שידורים חוזרים, מה שגורם לעיכובים. איבוד מנות יכול להיגרם עקב עומס ברשת, חומרה תקולה או שגיאות בקישור.
- RTT גבוה: RTT גבוה 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 בקרת העומס של TCP להפחית את קצב השידור.
- בעיות בהגדלת חלון TCP: ברשתות 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 ב- (הגודל המירבי של חבילות) MTU או פיצול חבילות, עלולים לגרום לאי-יעילות בהעברת הנתונים, במיוחד אם גודל המנות חורג מהמקסימום המותר (MTU) ויש לפצלן.

שלבי פתרון בעיות:

1. בדיקת RTT ושיהוי: ניתן להשתמש בפקודות ping או traceroute כדי לבדוק שיהוי ולאתר מקטעים ברשת עם זמן תגובה גבוה.
2. ניתוח גדלי חלונות TCP: ניתן להשתמש בכלי ניטור רשת כדי לבדוק את גודל חלון ה-TCPבחבילות. אם גודל החלון של המקבל קטן באופן עקבי, ייתכן שזה מצביע על צוואר בקבוק בצד המקבל.
3. בדיקת איבוד מנות: ניתן להשתמש בכלים כמו tcpdump או Wireshark כדי לזהות שידורים חוזרים או ACK כפולים, מה שעשוי להצביע על איבוד מנות.
4. בדיקת אלגוריתמי בקרת עומס: לוודא שמשתמשים באלגוריתמי בקרת עומס מתאימים (כגון CUBIC בלינוקס), במיוחד ברשתות מהירות.
5. בדיקת בעיות MTU: ניתן להשתמש בפקודת ping עם האפשרות M do- כדי לבדוק בעיות בזיהוי MTU במסלול הרשת.

6. ניטור עומס רשת: ניתן להשתמש בכלי ניטור רשת כדי לבדוק ניצול גבוה של רוחב פס או איבוד מנות בנתבים ובמתגים.

2. הבנת בקרת הזרימה בTCP:

בקרת הזרימה נועדה למנוע מהשולח להציף את המקבל בכמות נתונים שהוא אינו יכול לעבד. מנגנון זה מתבצע באמצעות חלון ה TCP- של המקבל, (rwnd) הקובע כמה נתונים לא מאושרים יכולים להיות בתעבורה בכל רגע נתון.

השפעה על הביצועים כאשר השולח מהיר יותר מהמקבל:

- צוואר בקבוק בצד המקבל: אם השולח בעל כוח עיבוד גבוה משמעותית ויכול לשלוח נתונים מהר יותר ממה שהמקבל יכול לעבד ,חלון ה TCP-של המקבל יתמלא במהירות. כאשר זה קורה, השולח ייאלץ להשהות את ההעברה עד שיתפנה מקום בזיכרון החוצץ של המקבל.
- תקופות השבתה של השולח: השולח ייכנס לעיתים קרובות למצבים של המתנה לקבלת אישורים (ACKs) שמעידים כי המקבל מוכן לקבל עוד נתונים, מצב זה גורם לניצול לא יעיל של משאבי השולח.
- ניצול לא אופטימלי של רוחב הפס: גם אם השולח יכול להעביר נתונים בקצב גבוה יותר, בקרת הזרימה תגביל את קצב ההעברה כך שיתאים ליכולת של המקבל, מה שעלול לגרום לביצועים נמוכים מהפוטנציאל.
- השהיות גבוהות בהעברת קבצים גדולים: בהעברת קבצים גדולים, בקרת הזרימה עשויה להאריך את זמן ההעברה ,מאחר שהשידור צריך להסתגל באופן רציף למהירות של המקבל.

****דרכים להקטנת הפגיעה בביצועים:**

- שיפור ביצועי המקבל: הגדלת גודל החוצץ (buffer) בצד המקבל או שיפור יכולות העיבוד שלו כדי לאפשר עיבוד יעיל יותר של הנתונים הנכנסים.
- הגדלת חלון TCP : הפעלת TCP Window Scaling תאפשר חלונות גדולים יותר, מה שיצמצם את תדירות ההשהיות הנגרמות מבקרת הזרימה.
- שימוש בחיבורים מקבילים: פתיחת מספר חיבורי TCP במקביל יכולה לשפר את ניצול המשאבים של השולח, במיוחד במקרים של העברת קבצים או סטרימינג.

3.כיצד בחירת מסלול משפיעה על ביצועי הרשת:

- שיהוי ועיכובים: בחירת המסלול משפיעה על זמן הסבב (RTT), מסלולים קצרים יותר או פחות עמוסים מפחיתים שיהוי ,מה שמשפר את זמן התגובה של הרשת.
- ניצול רוחב פס: מסלולים מסוימים מציעים קיבולת רוחב פס גבוהה יותר ,מה שמאפשר העברת נתונים מהירה יותר, בחירת מסלול עם רוחב פס נמוך עלולה ליצור צוואר בקבוק.

- אמינות וגיבוי: פרוטוקולי ניתוב לוקחים בחשבון את יציבות המסלול, מסלול יציב עם פחות תקלות משפר את ביצועי הרשת ומפחית סיכון לניתוקים.
- עומס ואיזון עומסים: החלטות ניתוב יכולות לפזר תנועה בין מסלולים שונים כדי למנוע עומסים ולשפר ביצועים. ניתוב לא מאוזן עלול להעמיס על מסלולים מסוימים תוך השארת אחרים בתת-ניצול.

גורמים שיש לקחת בחשבון בהחלטות ניתוב:

- עלות קישור/מדדים: פרוטוקולי ניתוב משתמשים במדדים כמו מספר קפיצות (hop count), שיהוי, רוחב פס ואמינות כדי לקבוע את המסלול האופטימלי.
- פרוטוקולי ניתוב BGP (Border Gateway Protocol) OSPF (Open Shortest Path First) משתמשים באלגוריתמים שונים לקבלת החלטות ניתוב.
- איכות שירות(QoS) מסלולים מסוימים עשויים להעדיף סוגי תעבורה מסוימים)כגון שיחות VoIPאו שיחות וידאו (כדי להבטיח איכות שירות גבוהה.
- ניתוב מבוסס מדיניות: מדיניות ארגונית יכולה להשפיע על הניתוב, למשל העדפת ספקי אינטרנט זולים יותר או מסלולים מאובטחים יותר.

4. מהו MPTCP?

Multipath TCP (MPTCP) מאפשר לחיבור TCP יחיד להשתמש במספר מסלולים בו-זמנית, תוך ניצול ריבוי ממשקי רשת (כגון Wi-Fi, Ethernet, LTE) כדי לשפר תפוקה ואמינות.

יתרונות של MPTCP:

- הגדלת תפוקה: על ידי שילוב של מספר ממשקי רשת MPTCP ,משיג תפוקה מצרפית גבוהה יותר בהשוואה ל TCP רגיל המשתמש במסלול יחיד.
- אמינות וגיבוי משופרים: אם מסלול אחד נכשל MPTCP ,מעביר את התעבורה באופן שקוף למסלול חלופי ללא ניתוק החיבור.
- איזון עומסים יעיל: MPTCP מפזר את הנתונים דינמית בין מספר מסלולים בהתאם לעומס הרשת ולאיכות המסלול ,לשימוש אופטימלי במשאבים.
- הפחתת שיהוי: ניתן לבחור את המסלול המהיר ביותר עבור זרמי נתונים מסוימים ,מה שמפחית את זמן התגובה ומשפר את ביצועי הרשת.
- תמיכה טובה יותר בניידות: אידיאלי למכשירים ניידים שעוברים מרשת Wi-Fi לרשת סלולרית מבלי לאבד את החיבור.

5. גורמים אפשריים לאיבוד מנות בשכבת הרשת:

- עומס בנתבים: נתבים עמוסים עלולים להפיל מנות כאשר התורים שלהם מתמלאים.
- חומרה תקולה: כבלים פגומים, נתבים או מתגים תקולים עשויים לגרום לשגיאות ולמנות פגומות.

- לולאות ניתוב או תצורה שגויה: נתבים שהוגדרו בצורה שגויה עלולים לגרום למנות להסתובב ללא סוף עד שהן נזרקות.
- חוסר התאמה בMTU: בעיות ב MTU-עלולות לגרום לפיצול מנות או להפלתן אם התקנים באמצע המסלול אינם יכולים להתמודד עם גודל מנות גדול.

גורמים אפשריים בשכבת התעבורה:

בקרת עומס בTCP- איבוד מנות מפעיל את מנגנוני בקרת העומס של TCP, מה שמפחית את קצב השידור של השולח.

גודש בזיכרון חוצץ (Buffer Overflows): חוסר מקום בחוצץ של השולח או המקבל עלול לגרום להפלת מנות.

זמני המתנה ושידורים חוזרים: שיהוי גבוה או תנודות ברשת (jitter) עלולים לגרום לפקיעות זמן (timeouts), שמובילים לשידורים חוזרים ולתפיסה של איבוד מנות.

צעדים לפתרון בעיות איבוד מנות:

1. ניטור עומס ברשת: ניתן להשתמש בכלים כמו Wireshark או netstat כדי לאתר קישורים עמוסים ולחלק מחדש את התעבורה.
2. בדיקה והחלפת חומרה תקולה: ניתן לבדוק ולהחליף כבלים, מתגים או נתבים תקולים לפי הצורך.
3. אופטימיזציה של תצורת ניתוב: לוודא שהגדרות הניתוב נכונות ואין לולאות ניתוב.
4. התאמת הגדרות ה MTU: ניתן להשתמש בזיהוי MTU במסלול (Path MTU Discovery) כדי לקבוע גודל MTU מתאים ולהימנע מפיצול מנות.
5. הגדלת חוצצי זיכרון (Buffer Sizes): בנתב או בנקודת הקצה, ניתן להגדיל את קיבולת החוצץ כדי להתמודד עם גלי תנועה פתאומיים.
6. הפעלת מדיניות Qos: ניתן לתעדף תעבורה קריטית כגון VoIP כדי להבטיח משלוח אמין בזמן עומס.

חלק 2 (ניתוח המאמרים):

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

תרומות עיקריות של המאמר:

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

- ייצוג FlowPic: המרת נתוני זרם תעבורה לרשת (network flow) לייצוג תמונתי בשם FlowPic אשר לוכד מידע על גודל המנות (packets) וזמני ההגעה שלהן.
- שימוש ב CNNs לסיווג תעבורה: יישום רשתות נוירונים (CNNs) לסיווג תעבורה, במקום הסתמכות על מאפיינים סטטיסטיים מחולצים ידנית.
- דיוק סיווג גבוה: השגת למעלה מ-96% דיוק בזיהוי קטגוריות תעבורה, כולל תעבורה מוצפנת בVPN וב TOR.
- יכולת הכללה ליישומים חדשים: הוכחת יכולת המודל לסווג בהצלחה יישומים שלא נראו במהלך האימון, עם דיוק של 99.9%.
- שימור פרטיות: סיווג תעבורה ללא שימוש בתוכן עצמו ובכך הימנעות מפגיעה בפרטיות.

מאפייני התעבורה והשיפורים הייחודיים:

המאמר מציג את , FlowPic אשר מבוסס על:

- גודל מנות
- זמן הגעת מנות
- התפלגות מנות לאורך זמן: ייצוג כ2D Histogram.

חידושים עיקריים:

- ייצוג תמונתי של זרם תעבורה: במקום שימוש במאפיינים סטטיסטיים מסורתיים, השיטה יוצרת תמונה (היסטוגרמה דו ממדית) המבוססת על גדלי מנות וזמני הגעה, ומאפשרת ל CNNsלסווג את התעבורה בצורה ויזואלית.
- חוסר תלות בתוכן החבילה: בניגוד לשיטות מסורתיות כמו DPI (Deep Packet Inspection), FlowPicאינו מסתמך על תוכן המטען, ולכן הוא אפקטיבי גם עבור תעבורה מוצפנת.

- ניתוח חלון זמן קצר: המערכת מסוגלת לסווג זרמי תעבורה קצרים בכיוון אחד בלבד, ללא צורך בניתוח זרם דו-כיווני מלא.

תוצאות עיקריות ותובנות:

דיוק בסיווג:

משימה	דיוק
סיווג תעבורה ללאVPN	85.0%
סיווג תעבורה מוצפנת בVPN-	98.4%
סיווג תעבורה מוצפנת בSSL/TLS-	67.8%
סיווג תעבורה מוצפנת ב VPN-כאשר האימון בוצע על נתונים ללא VPN	99.4% - 78.9%
זיהוי יישומים	99.7%

תובנות מרכזיות:

- שיטת FlowPic המבוססת על CNNs משיגה ביצועים טובים יותר בהשוואה לשיטות מסורתיות של למידת מכונה.
- המודל מזהה קטגוריות תעבורה גם כאשר נעשה שימוש בהצפנה כמו VPN ו TOR
- גם כאשר מסירים יישומים מסוימים מהאימון, המודל עדיין מצליח לסווג אותם נכון, מה שמעיד על חוסן גבוה.
- השיטה מציגה דיוק גבוה משמעותית בהשוואה למחקרים קודמים, במיוחד בזיהוי יישומים.

איורים מרכזיים ותובנות חזותיות:

1. דוגמאות FlowPics ליישומי וידאו: מציג כיצד יישומים שונים כגון Netflix ו Skype יוצרים תבניות תעבורה ייחודיות.
2. השוואת קטגוריות תעבורה תחת הצפנות שונות: מדגים כיצד תעבורת VoIP וצ'אט משתנה כאשר נעשה שימוש ב VPNו TOR
3. ארכיטקטורת ה CNN - בהשראת LeNet-5 : איור של שכבות הקונבולוציה המשמשות לסיווג FlowPic.
4. מטריצות בלבול: השוואת דיוק הסיווג בין תעבורה רגילה VPN,TOR .

<u>הסבר האלגוריתם המוצג במאמר למען ההקשר (flowpic+CNN):</u>

המאמר מציע שימוש ב – CNNs-טכניקה נפוצה בזיהוי תמונות – לסיווג תעבורת אינטרנט מוצפנת. הרעיון המרכזי הוא שתבניות תעבורה ניתנות לייצוג כתמונות ,(FlowPics) מה שמאפשר ל CNN לזהות ולסווג אותן באופן דומה לזיהוי אובייקטים בתמונות.

כיצד CNNs משמשים בסיווג FlowPics:

1. המרת נתוני הרשת לFlowPics

- גודל המנות וזמני ההגעה מתורגמים לתמונה בגודל 1500×1500 פיקסלים בגווני 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

- הCNN מזהה תבניות בתמונה ומסווג את סוג התעבורה.

3. שימוש ב CNN-לסיווג:

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

<u>הסבר על מבנה האלגוריתם:</u>

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

מבנה הרשת:

- שכבת קלט FlowPic בגודל 1500×1500 מוזן לCNN-
- שכבות קונבולוציה:

- CONV1: 10 מסננים בגודל 10×10 עם צעד של 5 , מפיק 10 מפות מאפיינים בגודל 300×300.
- Max Pooling, מפחית את הגודל ל-150×150.
- CONV2: 20 מסננים בגודל 10×10 עם צעד של 5, מפיק 20 מפות מאפיינים בגודל 30×30.
- Max Pooling מפחית את הגודל ל-15×15.

- שכבת Fully Connected עם 64 נוירונים.
- שכבת פלט עם Softmax (פונקציית הפעלה אשר ממירה את ערכי הפלט לווקטור של הסתברויות שמסתכמות ל-1). לסיווג קטגוריות התעבורה.

<u>יתרונות השימוש ב CNN-בסיווג תעבורה מוצפנת:</u>

- לומד מאפיינים אוטומטית, ללא צורך בהגדרת מאפיינים ידנית.
- מאפשר סיווג תעבורה גם כשהיא מוצפנת.
- משיג ביצועים גבוהים יותר משיטות מסורתיות כמו SVM, KNN
- מסוגל להכליל ולזהות יישומים שלא נראו במהלך האימון.

מסקנות עיקריות מהמאמר:

1. <u>שיטה חדשנית ומדויקת לסיווג תעבורה מוצפנת:</u> השיטה מצליחה לסווג קטגוריות תעבורה (גלישה, צ'אט, וידאו ,VoIP ,העברת קבצים) בדיוק של מעל 96%, תעבורה שעוברת דרך VPN מסווגת בדיוק של 99.2%, ודרך Tor בדיוק של מעל 89%, למעט העברת קבצים.
2. <u>יכולת הכללה על אפליקציות חדשות:</u> גם כאשר האפליקציה הספציפית לא הופיעה באימון, ניתן עדיין לסווג את התעבורה לקטגוריה הנכונה, למשל, המודל אומן ללא נתוני *Facebook Video* אך הצליח לזהות אותו כווידאו בדיוק של 99.9%
3. <u>עדיפות על שיטות קודמות:</u> השיטה יעילה יותר מהשיטות הקלאסיות של למידה מפוקחת המשתמשות בתכונות מהונדסות כגון *SVM*, *KNN* בשונה מניתוח חבילות עמוק (DPI), השיטה אינה פולשנית לפרטיות ומתבססת רק על גודל וזמן ההגעה של החבילות.
4. <u>התמודדות עם הצפנה(VPN, Tor):</u> השיטה מצליחה לסווג תעבורה למרות הצפנה , בזכות שימוש במידע זרימה בלבד, זיהוי סוגי הצפנה מתבצע בדיוק של 88.4%, כאשר Tor מזוהה בדיוק של 97.7%.
5. <u>פשטות ויעילות של מודל הCNN-</u> הארכיטקטורה מבוססת *LeNet-5* עם שכבות קונבולוציה, מיקוד (Max Pooling) ושכבות Fully Connected , השיטה פועלת ללא התאמות פרטניות לכל סוג תעבורה ,מה שמאפשר שימוש חוזר באותה ארכיטקטורה לכל משימות הסיווג.
6. <u>אפשרויות לשיפור עתידי:</u> ניתן לשפר את ביצועי המודל ע"י התאמות בהיפר-פרמטרים, למשל הקטנת רזולוציית *FlowPic* מ-1500×1500 ל-300×300, 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 הצפנת TLS 1.3 ושימוש ב Encrypted ClientHello (ECH), התרומות המרכזיות כוללות:

- מאגר נתונים רחב-היקף חדש: המאמר מציג מאגר נתונים מגוון הכולל יותר מ-600,000 זרמי TLS, שנאספו מאזורים שונים בעולם (צפון אמריקה, אירופה ואסיה). מאגר זה הוא אחד ממאגרי הנתונים הפומביים הגדולים ביותר לסיווג תעבורה.
- מסווג תעבורה היברידי מבוסס Random Forest (hRFTC) : המאמר מציג מסווג חדשני המשלב מטא-נתונים לא מוצפנים של TLS, כגון key share, רשימת הצפנים והרחבות עם מאפיינים סטטיסטיים מבוססי זרם (כגון זמני הגעה בין מנות והתפלגות גדלי מנות).
- הערכת שיטות קיימות לסיווג תעבורה: המחקר משווה מסווגים מתקדמים (מבוססי מנות ומבוססי זרמים) על תעבורה מוצפנת ב ClientHello, מזהה את חולשותיהם ומציע שיפורים.
- תובנות על שונות גאוגרפית: המאמר מראה כי מסווגים שאומנו באזור גאוגרפי מסוים אינם מתפקדים היטב באזורים אחרים, ודורשים אימון מחדש כדי להגיע לביצועים אופטימליים.

מאפייני התעבורה והשיפורים הייחודיים:

המסווג החדשני hRFTC שמשלב:

- מאפיינים מבוססי מנות : שדות הקשורים לשלב ההתחלתי של חיבור TLS כגון רשימות הצפנים והרחבות.
- מאפיינים מבוססי זרם: סטטיסטיקות של גודל מנות, זמני הגעה של מנות, ותכונות מבוססות רצפים.

חידושים עיקריים

- שימוש במטא-נתונים של TLS לסיווג: למרות שהצפנת ECH מסתירה חלק מהמידע, עדיין קיימים שדות בלתי מוצפנים שיכולים לשמש לסיווג. המסווג מנצל שדות אלו בצורה אופטימלית.

- מאפיינים מבוססי זרם עבור סיווג מוקדם: במקום להמתין לקבלת אלפי מנות (כפי שנדרש בשיטות מסורתיות) hRFTC ,מצליח לסווג תעבורה באמצעות מספר קטן של מנות ראשוניות בלבד.
- שילוב היברידי לשיפור הדיוק :השילוב בין מאפיינים מבוססי מנות וזרם משפר משמעותית את הביצועים בהשוואה למסווגים טהורים המבוססים על אחת מהשיטות בלבד.

תוצאות עיקריות ותובנות:

<u>ביצועי שיטות סיווג שונות:</u>

שיטה	Macro F-Score
hRFTCהשיטה המוצעת	94.6%
UWמודלUniversity of Waterloo	86.3%
hC4.5מבוסס עץ החלטה	79.4%
CESNETמסווג מבוסס זרם	72.1%
MATECמודל רשת נוירונים	38.4%
BGRUAרשת GRU עם מנגנון תשומת לב	35.7%
RB-RF Random Forestמבוסס מנות	39.2%

מסקנות עיקריות:

- hRFTC משיג ביצועים טובים יותר מכל שאר המסווגים, כולל רשתות נוירונים מתקדמות.
- מסווגים מבוססי מנות נכשלים (דיוק של 38.4%) כאשר נעשה שימוש בהצפנת ECH, אך שילוב עם מאפיינים מבוססי זרם מאפשר להגיע לדיוק של 94.6%.
- מודלים שנאומנו באזור אחד אינם מתפקדים היטב באזור אחר ,בשל הבדלים בהגדרות TLS ודפוסי תעבורה.

<u>הסבר על האלגוריתם המדובר במאמר למען ההקשר (hrftc):</u>

hRFTC (Hybrid Random Forest Traffic Classifier) הוא אלגוריתם מתקדם לסיווג מוקדם של תעבורה ,(eTC) המשלב בין מאפיינים מבוססי מנות וזרם כדי להתמודד עם הצפנת Encrypted ClientHello (ECH)

<u>שלבי העבודה שלhRFTC</u>

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 מבוססי מנות עם נתונים מבוססי זרם לווקטור מאפיינים אחד.
- נרמול המאפיינים והתמודדות עם נתונים חסרים.

שלב 3: סיווג עם Random Forest

- שימוש באלגוריתם Random Forest לסיווג תעבורה בזמן אמת.

<u>יתרונות השימוש ב Random Forest-בhRFTC-</u>

- עמידות לרעש :בניגוד לרשתות נוירונים, המודל אינו רגיש יתר על המידה לשינויים קטנים ברצפי מנות.
- ביצועים מהירים :המסווג יכול לעבד 100,000זרמים בשנייה ,מה שהופך אותו למתאים לשימוש בזמן אמת.
- יכולת זיהוי של קשרים מורכבים בין מאפיינים מבוססי מנות וזרם.

מאמר Analyzing HTTPS Encryped Traffic to IDE:

המאמר מציג שיטה חדשנית לזיהוי מערכת ההפעלה, הדפדפן והיישום של משתמש על בסיס ניתוח תעבורת HTTPS מוצפנת בלבד.

התרומות המרכזיות הן:

- המחקר הראשון לסיווג מערכת הפעלה, דפדפן ויישום מתעבורת HTTPS : מראה כי יריב פסיבי יכול להסיק את מערכת ההפעלה, הדפדפן וסוג היישום למרות הצפנת HTTPS, מדגים כי ניתן לבצע סיווג על סמך תכונות שמופקות ממטא-נתונים של TLS/SSL והתנהגות רשתית.
- סט תכונות חדש לשיפור הסיווג: מציע סט חדש של תכונות הקשורות להתנהגות "מתפרצת" בתעבורה (פיקים של פעילות), מציג תכונות של לחיצת יד TLS/SSL המאפשרות הבחנה בין דפדפנים ויישומים, משיג דיוק של 96.06% בסיווג, שיפור לעומת 93.51% בבסיס.
- מסד נתונים רחב היקף: מספק מסד נתונים של מעל 20,000 ssנים מסומנים, הכוללים סוגים שונים של מערכות הפעלה) ווינדוס, לינוקס-אובונטו ,macOS) , דפדפנים (כרום, פיירפוקס, ספארי, אקספלורר) ויישומים (יוטיוב, פייסבוק, טוויטר) מסד הנתונים פתוח למחקר נוסף.
- מודל סיווג מדויק במיוחד: משתמש במכונת וקטורים תומכת (SVM) עם קרנל RBF , מעריך מספר סטים של תכונות ומראה כי שילוב התכונות הבסיסיות עם התכונות החדשות נותן את התוצאה הטובה ביותר.

תכונות תעבורה בשימוש:

1.תכונות בסיסיות (נפוצות בסיווג תעבורה):

תכונות מבוססות חבילות

- מספר החבילות שנשלחו קדימה/אחורה.
- סך כל הבתים שנשלחו/התקבלו.
- זמני הגעה מינימליים, מקסימליים וממוצעים.
- סטיית תקן של גודל החבילה ושל זמני ההגעה.
- סך כל מספר החבילות, יחס הכיווניות, וערכי TTL

תכונות חדשות שהוצגו במאמר:

תכונות חדשניות 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 "פיקים" בתעבורה – התפרצויות של נתונים עם תקופות שקט ביניהן.
  - זיהוי יישומים שונים לפי מבנה התעבורה שלהם, משמש להבדלה בין דפדפנים לפי דפוסי הפיקים הייחודיים שלהם.

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

תוצאות עיקריות ותובנות

<u>דיוק הסיווג</u>

סט תכונות	**דיוק סיווג יישום**	**דיוק סיווג דפדפן**	**דיוק סיווג מערכת הפעלה**
תכונות בסיסיות	92.3%	93.51%	93.52%
תכונות חדשות בלבד	93.7%	94.2%	94.5%
שילוב בסיס + תכונות חדשות	96.06%	96.06%	96.06%

נתונים חשובים ותובנות ויזואליות

הרכב מסד הנתונים

- **מערכות הפעלה** :ווינדוס, אובונטו.macOS ,
- **דפדפנים** :כרום, פיירפוקס, ספארי, אינטרנט אקספלורר.
- **יישומים** :יוטיוב, פייסבוק, טוויטר, דרופבוקס.TeamViewer ,

חשיבות תכונות

- תכונות TLS מסייעות להבחין בין דפדפנים וסוגי מערכות הפעלה.
- תכונות מבוססות זמנים של חבילות משפרות את סיווג היישומים.

גרף שיפור הדיוק

- תכונות בסיס בלבד משיגות דיוק של ~93%.
- הוספת תכונות פיקים ומטא-נתונים של TLS משפרת את הדיוק ל-96.06%.

מסקנות עיקריות מהמאמר

- גם אם הנתונים מוצפנים ב HTTPS,-ניתן להסיק מידע על המשתמש
  - ניתן לזהות את מערכת ההפעלה, הדפדפן והיישום באמצעות ניתוח מטא-נתונים של TLS ומבנה התעבורה.
  - תוקפים יכולים לעקוב אחר משתמשים ולזהות פעילותם גם ללא גישה לנתונים עצמם.
- תוספת התכונות חדשות משפרת את דיוק הסיווג
  - שימוש בתכונות מבוססות TLS והתנהגות מתפרצת (bursty behavior) מעלה את דיוק הסיווג ל-96.06%.
  - שילוב נתוני TLS ותזמון חבילות מסייע בזיהוי מדויק של הדפדפן והיישום.
- ניתן לזהות דפדפנים ויישומים לפי דפוסי שליחת הנתונים שלהם
  - דפדפנים שונים (Chrome, Firefox, Safari) יוצרים דפוסי זרימה ייחודיים ברשת.

- אתרים כמו YouTube ו Facebook-שולחים נתונים באופן שמאפשר להבדיל ביניהם.

- <u>השלכות על פרטיות – ניתן לזהות משתמשים גם ללא חשיפת הנתונים</u>
  - תוקפים יכולים להשתמש במידע זה כדי לעקוב אחרי משתמשים ולבצע התקפות ממוקדות.
  - רשתות פרסום יכולות לנצל זאת כדי לזהות דפדפנים ויישומים לצורך מעקב והתאמת פרסומות.
- <u>כיצד ניתן לשפר פרטיות?</u>
  - שימוש ב VPN-ובהשהיית חבילות עשוי למנוע זיהוי המבוסס על דפוסי תעבורה.
  - פרוטוקולים כמו ECH (Encrypted Client Hello) יכולים להצפין מידע מטא-דאטה ולמנוע זיהוי של הדפדפן והאפליקציה.

חלק 3(ניתוח הגרפים ומסקנות):

טבלת המידע המייצגת את קבצי הpcap השונים:

PCAP DataFrame

Pcap file	Flow size	Flow Volume (bytes)	Flow duration (second:	Avg Packet size (bytes)	Avg Packet IAT (secon	CV IAT	Unique Flows	Flow Directionality Rati	Http Count	Tcp Flags	Ip protocols
1chrome regular.pcap	32596	32973434	178.73	1011.58	0.005483	21.770511	647	0.79	HTTP2-4646 HTTP3-44	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 HTTP3-20	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 HTTP3-539	ACK-1153 FIN-70 SYN-	17-7384 1-15 6-486
4firefox spotify.pcap	10978	12288540	186.83	1119.38	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 HTTP3-859	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-984 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

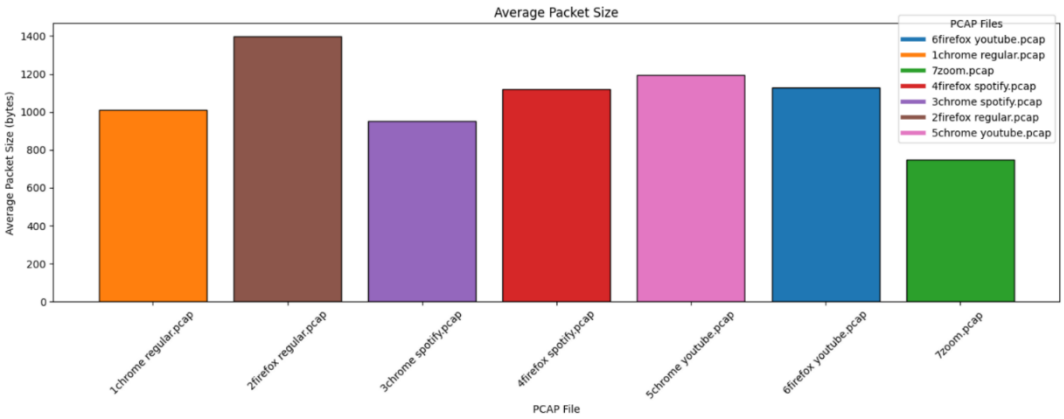
The name of the loaded PCAP file.

הצגת הגרפים מהקוד איתם השווינו בין התעבורות השונות :

קודם כל למען ההשוואה נציין, כי לכל ההקלטות המתועדות כאן זמן ההקלטה הוא בקירוב 3 דקות, בשביל לנרמל את ההשוואה בין כל התעבורות על גבי זמן, וגם כדי שנוכל לזהות דפוסים מסוימים שבהקלטות קצרות יותר יכול להיות והיו מתפספסים.

1.Average Packet Size גודל חבילה ממוצעת:

הגרף מציג את גודל החבילה הממוצע עבור קובצי PCAP שנאספו, ומאפשר לזהות דפוסים ברורים בהתנהגות התעבורה של יישומים שונים.



מסקנות מהגרף:

1. גודל חבילה ממוצע של firefox גדול מchrome.

- הגורם לכך הוא שFirefox נוטה לבצע פחות פיצול של חבילות TCP לעומת chrome.
- מנגנון ניהול TCP של firefox מאחד יותר נתונים לפני שליחת חבילה, בניגוד לchrome ששולח חבילות קטנות יותר באופן תדיר.

- Chrome מבצע שליחה מבוזרת ומהירה יותר של חבילות קטנות ,בעוד Firefox ממקסם את גודל החבילה (MTU) במקרים מסוימים, מה שמוביל לגודל חבילה ממוצע גבוה יותר.

2. <u>גודל חבילה ממוצע של זום קטן מכל שאר הגלישות.</u>

- Zoom משתמש בudp (במקום TCP) להעברת חבילות מדיה בזמן אמת, מה שמקטין את הצורך באיחוד נתונים ושולח חבילות קטנות בקצב מהיר.
- פרוטוקול (Real-time Transport Protocol) RTP שמשמש את Zoom בנוי לשלוח מידע בתדירות גבוהה מאוד אך עם חבילות קטנות ,כדי למנוע השהיות בקצה המקבל.
- גודל חבילה קטן מסייע להימנע מתיקוני שגיאות כבדים שעלולים לגרום לעיכוב בשידור.

3. <u>גדלי חבילה ממוצעים של YouTube ו-Spotify דומים מאוד בין שני הדפדפנים.</u>

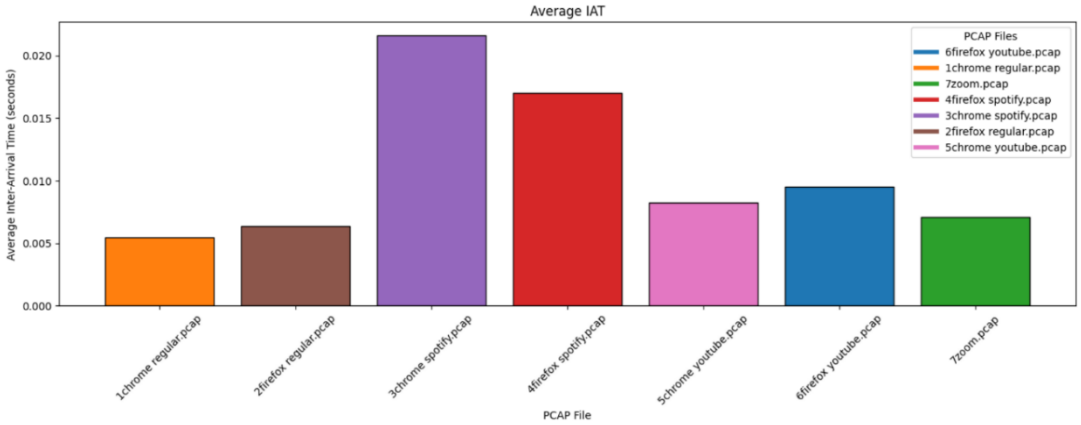
- YouTube ו-Spotify משתמשים בפרוטוקולי HTTP מבוסי DASH/HLS , המאפשרים הורדת מקטעי מדיה גדולים בגודל קבוע פחות או יותר.
- בשני הדפדפנים, תעבורת המדיה מגיעה מהשרתים באותו אופן, והגודל הממוצע של חבילות הנתונים נשמר בסטייה מינימלית.
- שני השירותים משתמשים ב- buffering וב- adaptive bitrate streaming ולכן החבילות שלהם דומות מבחינת 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 שנקלטת במהלך תעבורת הרשת של כל אחת מהאפליקציות/הדפדפנים שנבדקו.

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

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



מסקנות מהגרף:

<u>IAT ממוצע כמעט זהה בגלישת דפדפן רגילה בין שני הדפדפנים.</u>

- ניתן לראות מהגרף כי Chrome (regular) ו- Firefox (regular) מציגים ערכי IAT דומים מאוד. הסיבה לכך היא ששני הדפדפנים מבצעים בקשות HTTP באופן דומה, בהתבסס על פרוטוקול TCP, שבו מנגנוני הזרימה של הנתונים עובדים בצורה קרובה מאוד.
- TCP מבצע ניהול Congestion Control דומה בשני הדפדפנים ,כך שהמרווח בין החבילות נשאר יציב.

<u>IAT הממוצע של Spotify גדול משמעותית בהשוואה לשאר הגלישות.</u>

- Spotify (בשני הדפדפנים) מציג IAT גבוה מאוד בהשוואה לשאר היישומים. הסיבה לכך היא שהזרמת מוזיקה ב- Spotify משתמשת ב- buffering אינטנסיבי,

האפליקציה מורידה מקטעי אודיו גדולים מראש (pre-fetching) ולכן אין צורך בשליחה תכופה של חבילות נתונים.

- בגלל buffering , חבילות גדולות נשלחות בפרקי זמן יחסית רחוקים זה מזה, מה שגורם לעלייה בIAT-.

<u>IAT ממוצע של יוטיוב בינוני לשאר הגלישות</u>

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

<u>IAT ממוצע של זום והדפדפנים קטן יחסית לשאר הגלישות:</u>

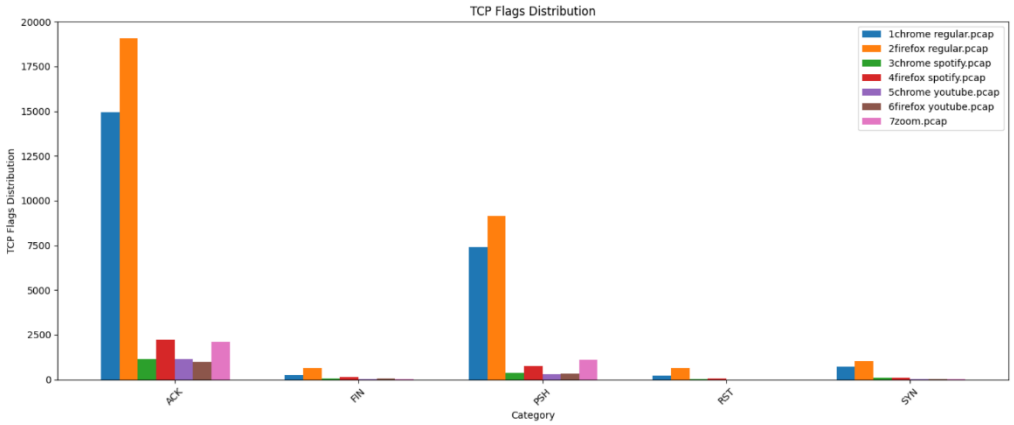
- Zoom מציג IAT ממוצע נמוך ביותר- מה שמצביע על שליחה רציפה ומהירה של חבילות קטנות.
- Zoom משתמש ב- UDP ובפרוטוקול RTP (Real-time Transport Protocol) , שמטרתו להקטין השהיות ולהבטיח שהשידור בזמן אמת לא יושפע מתקלות רשת.
- הגלישה הרגילה ב Chrome וב- Firefox גם היא מציגה IAT נמוך יחסית, מכיוון שהדפדפנים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

3. TCP Flags Distribution:

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

דגלי TCP הם סיביות (flags) בכותרת של חבילת TCP, אשר מציינים את מצב החיבור והתנהגות הזרימה של התעבורה בין לקוח לשרת.

מניתוח הגרף ניתן להסיק מסקנות לגבי אופי הגלישה, שימוש בפרוטוקולים שונים, ודפוסי התעבורה של כל יישום.



מסקנות מהגרף:

1. ישנה כמות גדולה של ACK לעומת PSH בכל סוגי הגלישות.

- ניתן לראות בגרף כי מספר החבילות עם דגל ACK (Acknowledgment) גבוה משמעותית בהשוואה ל- PSH.
- הסיבה לכך היא שמנגנון ה TCP- מחייב אישור (ACK) על כל חבילה 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. בגלישת הדפדפנים הרגילה יש יותר SYN מאשר בשאר הגלישות.

- SYN הוא דגל שנמצא רק בחבילות הראשונות של חיבור TCP חדש כחלק מתהליך לחיצת היד המשולשת.
- בגלישה בדפדפן יש ביקורים תכופים באתרים רבים , כך שכל אתר חדש מחייב יצירת חיבור TCP חדש , ולכן יש יותר חבילות SYN.

- לעומת זאת, בשירותי סטרימינג כמו Spotify, YouTube Zoom- החיבור נשמר לאורך זמן ולכן כמות ה SYN- נמוכה יותר.

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

- RST (reset) מופיע בעיקר כאשר חיבור נסגר באופן חריג, והוא פחות שכיח.
- FIN מופיע כאשר צד אחד סוגר את החיבור בצורה מסודרת, אך בתעבורת אינטרנט מודרנית, דפדפנים ויישומים נוטים לשמור חיבורים פתוחים כל עוד ניתן, ולכן יש מעט חבילות עם דגל זה.
- SYN מופיע רק בתחילת חיבור ,TCP ולכן הכמות שלו נמוכה יחסית לכלל החבילות.

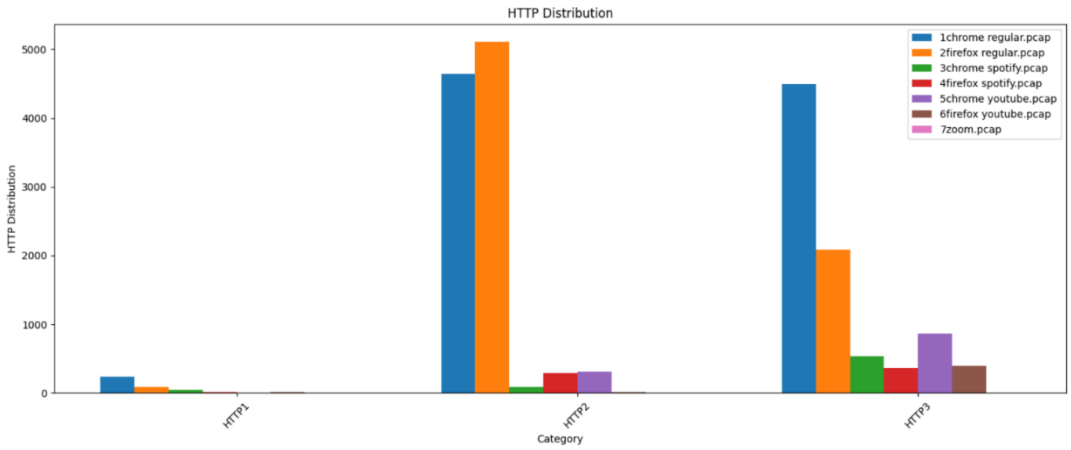
4. בגלישת דפדפנים רגילה מכל סוג, יש יותר מכל סוגי ה- Flags מאשר בשאר סוגי הגלישות.

- בגלישה בדפדפנים יש הרבה מאוד בקשות HTTP/S שמתבצעות דרך TCP ולכן יש יותר שימוש בכל סוגי הדגלים בהשוואה לשירותים מבוססי סטרימינג (שבהם יש פחות חיבורים חדשים).
- גלישת דפדפן יוצרת המון חיבורים קצרים לאתרים שונים, לעומת סטרימינג שבו יש חיבור ממושך אחד ,מה שמסביר את השכיחות הגבוהה של SYN, PSH, FIN וACK.

5. Zoom , youtube ו spotify אינם דומיננטיים בגרף זה עקב העדפתם לפרוטוקול udp להעברת המידע.

HTTP Distribution.4 :

הגרף מציג את התפלגות פרוטוקולי HTTP (HTTP1, HTTP2, HTTP3/QUIC) עבור סוגי תעבורה שונים שנלכדו בקובצי ה.PCAP-
פרוטוקולים אלה מייצגים את הדרך שבה הדפדפנים והאפליקציות מתקשרים עם השרתים ומספקים מידע על ההעדפה של כל דפדפן ויישום לפרוטוקול מסוים.



מסקנות מהגרף:

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

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

2. לגלישות דפדפן יש כמות רבה יותר באופן מובהק מאשר לשאר הגלישות עקב כך שישנו ביקור בכמות אתרים לעומת אתר אחד.

- גלישת דפדפן מייצרת הרבה חיבורי HTTP שונים, מאחר והמשתמש עובר בין מספר אתרים שונים.
- בשירותים כמו YouTube, Spotify ו Zoom,-יש חיבור ממושך אחד, ולכן יש פחות בקשות HTTP יחסית.

- הגרף אכן מראה כמות גדולה משמעותית של חבילות HTTP בגלישה רגילה בהשוואה לשאר היישומים.

3. <u>כאשר פיירפוקס משתמש ביוטיוב, כמעט ואינו משתמש בחבילות HTTP2 למרות העדפה הברורה שלו ל TLS,-וזאת מכיוון שיוטיוב משתמש בשרתי גוגל להעברת המידע.</u>

- YouTube פועל על שרתי Google אשר מעדיפים QUIC (HTTP3)
- Firefox נאלץ להשתמש בפרוטוקול המועדף על ,(HTTP3) YouTube גם אם הוא עצמו מעדיף HTTP2
- הנתונים בגרף תומכים בכך: מעט מאוד חבילות HTTP2 נמצאות בתעבורת YouTube בפיירפוקס.

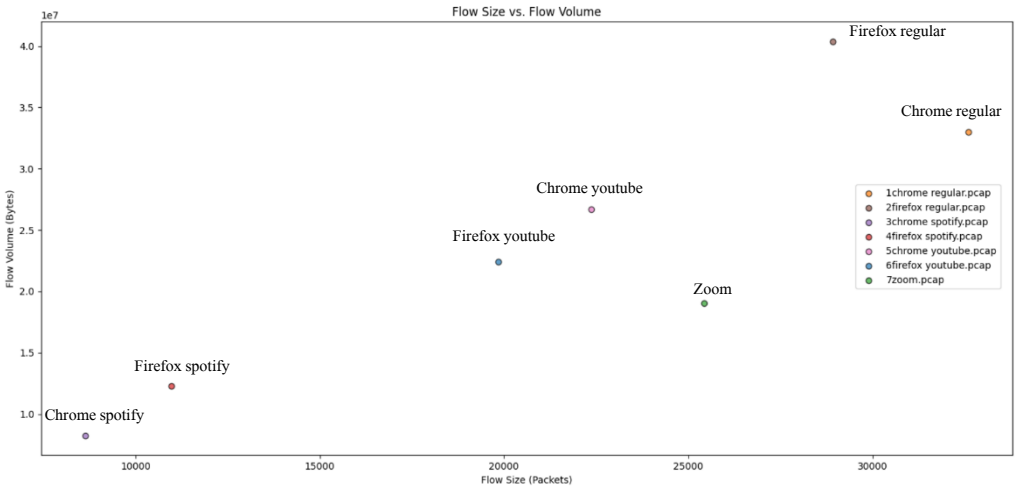
4. <u>Zoom בעקבות הפרוטוקול שלו DTLS-SRTP לא מאפשר לפענח את ההקלטות.</u>

- Zoomמצפין את התעבורה שלו באמצעות ,DTLS-SRTP ולכן אינו משתמש ב-HTTPלהעברת המדיה.
- כתוצאה מכך ,כמעט ואין חבילות HTTP בתעבורת ,Zoom מאחר וכל הנתונים המועברים מוצפנים ואינם מזוהים כ HTTP רגיל.
- הגרף אכן מציג נוכחות מינימלית של חבילות HTTP בתעבורת zoom.

:Flow size vs Flow volume.5

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

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



מסקנה מהגרף:

לכל סוג תעבורה יש פלח משלו בגרף:

ההבדלים המשמעותיים בין סוגי התעבורה:

1. **YouTube** צורך נפח נתונים גדול אך עם **כמות חבילות בינונית**, לשאר, מכיוון שהוא משדר **וידאו + אודיו** , כאשר הווידאו דורש הרבה יותר משאבים. הוא משתמש ב- *Adaptive Streaming* (DASH), שמחלק את הסרטון **למקטעים גדולים** ולכן כמות החבילות היא בינונית – לא קטנה כמו ב Spotify - ולא עצומה כמו בZoom

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 וידאו כבד או מידע ויזואלי להבדיל מ YouTube ו zoom.

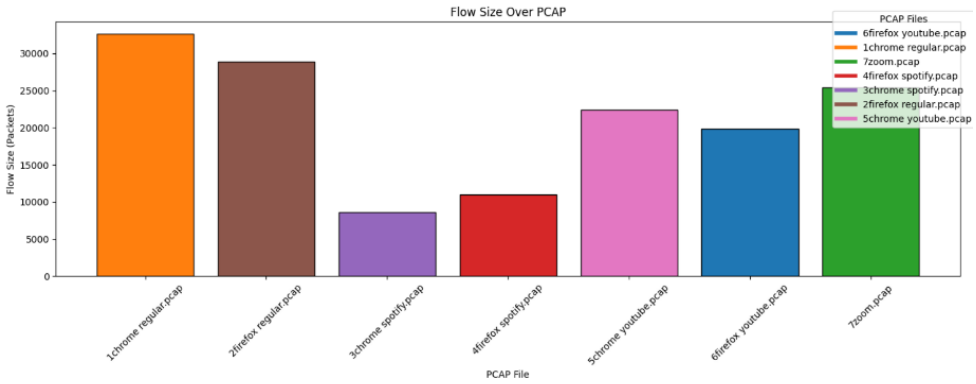
Zoom.3 המבוסס על UDP, **שולח הרבה מאוד חבילות קטנות** ביחס לשאר על מנת לשמור על תקשורת רציפה, וכדי להעביר אודיו ווידאו בתווך התקשורת, ולכן יש לו Flow Sizeגבוה אך Flow Volume נמוך.

Flow Size Over PCAP.6 :

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

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

בהשוואה זו נתמקד בשלוש האפליקציות Zoom, YouTube ו Spotify,-תוך כדי התעלמות מגלישה רגילה עקב אופי התעבורה המשתנה שלה.



1. ל Zoom יש את כמות החבילות הגדולה ביותר, ל YouTube-כמות בינונית ול Spotify-כמות קטנה

Zoom שולח את מספר החבילות הגבוה ביותר

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

YouTube נמצא באמצע עם כמות חבילות בינונית

- YouTubeמבוסס על HTTP3/QUIC או HTTP2, כלומר משתמש ב-*buffering*חכם ובזרימה של חבילות בינוניות-גדולות.
- הסרטונים מועברים בקבוצות של חבילות גדולות ולכן הוא צורך פחות חבילות יחסית ל Zoom, אך יותר מ Spotify

Spotify שולח הכי מעט חבילות מבין השלושה

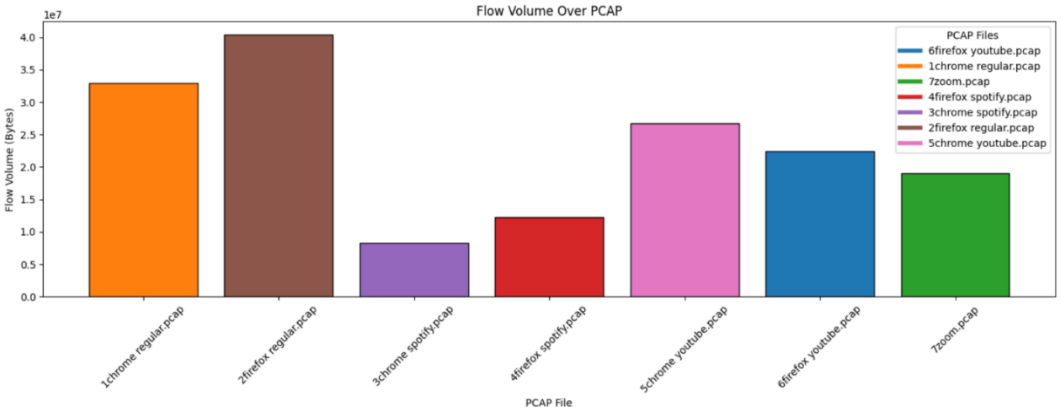
- Spotifyמשתמש באסטרטגיית *buffering* יעילה , בה הוא מוריד חלקים גדולים של השיר מראש, מה שמפחית את כמות החבילות הנשלחות בזמן אמת.
- לכן ,הוא צורך את כמות החבילות הקטנה ביותר מבין שלושת השירותים.

7. Flow Volume Over PCAP :

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

- **ציר X** – קובצי ה PCAP השונים, שכל אחד מהם מייצג סוג תעבורה.
- **ציר Y** – נפח התעבורה הכולל (בבייטים), כלומר כמה נתונים הועברו במהלך השימוש בכל אפליקציה.

בהשוואה זו נתמקד בשלוש האפליקציות Zoom, YouTube ו Spotify,-תוך כדי התעלמות מגלישה רגילה עקב אופי התעבורה המשתנה שלה.



Youtube צורך נפח נתונים גדול יותר מהשאר, Spotify הכי קטן, Zoom בינוני.

1. **YouTube צורך את נפח הנתונים הגדול ביותר**
   - **YouTubeמשדר וידאו + אודיו** ,ולכן נפח הנתונים גבוה בהרבה משירותי סטרימינג של אודיו בלבד.
   - הווידאו מועבר באיכות גבוהה, תוך שימוש בפרוטוקול **DASH** או **QUIC**, שגורם לטעינה של מקטעים גדולים בבת אחת.
   - לכן, נפח הנתונים של YouTube הוא הגדול ביותר מבין שלושת השירותים.
2. **Zoom נמצא באמצע עם נפח נתונים בינוני**
   - Zoom מעביר שיחות וידאו ואודיו, אך עושה זאת באמצעות חבילות קטנות בפרוטוקול UDP
   - נפח הנתונים אינו עצום כמו ב ,YouTube- אך עדיין גבוה יותר מ Spotify משום שהשירות משדר באופן רציף וללא דחיסה אגרסיבית.
   - לכן, נפח התעבורה של Zoom בינוני – יותר מ Spotify-אך פחות מ-YouTube.
3. Spotifyצורך את נפח הנתונים הקטן ביותר
   - Spotify משדר רק אודיו.

8.IP Protocols Distribution :

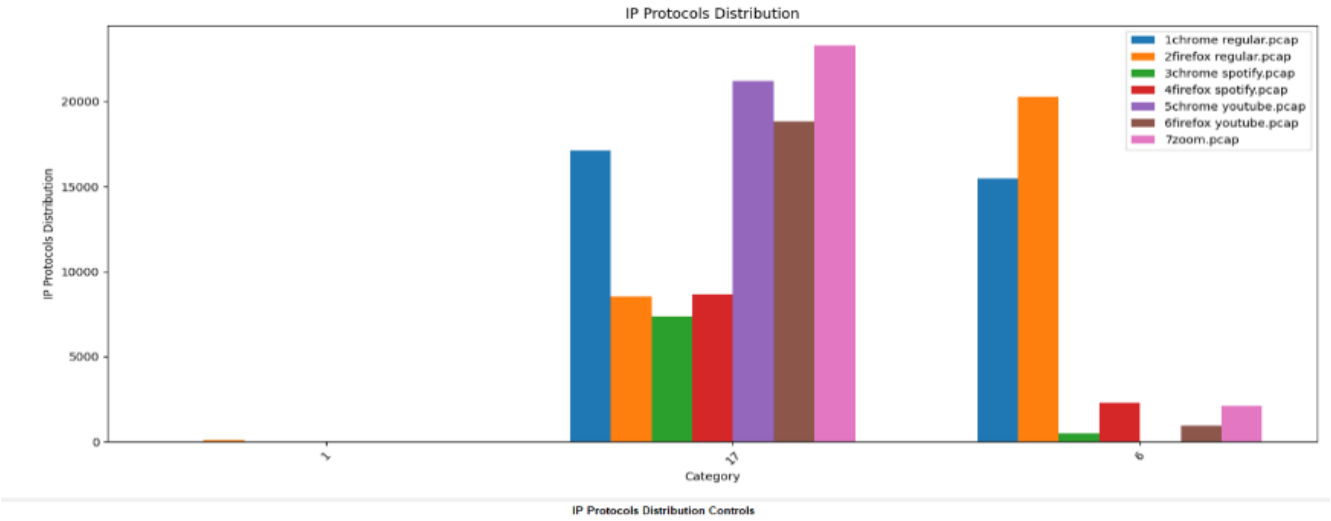
17 זה UDP

שש זה TCP

1 זה ICMP

הגרף מציג את התפלגות הפרוטוקולים ברמת ה IP כלומר כמה מהחבילות בכל קובץ PCAP השתמשו בכל פרוטוקול

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



מסקנות מהגרף:

Chrome משתמש יותר ב UDP-ו Firefox-משתמש יותר בTCP-

- Chrome אכן שולח יותר תעבורת UDP בהשוואה ל Firefox, - מה שמעיד על כך שהוא משתמש בפרוטוקול QUIC (HTTP3), שמבוסס על UDP
- Firefoxלעומת זאת, משתמש בעיקר ב TCP, -מה שמרמז על העדפתו ל HTTP2, - שעובד על TCP מעל TLS.

Youtube ו Zoom משתמשים ברוב מוחץ של udp לעומת tcp

- Zoom מבוסס על 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 והפחתת השהיות (Latency) בהזרמת וידאו.
- כמות ה TCP-קטנה יחסית, מאחר ורוב התעבורה מתבצעת **בהזרמת וידאו שמבוססת UDP**

Spotify משתמש ברוב מובהק של UDP לעומת TCP

- Spotify משתמש ב UDP-כדי לאפשר שידור אודיו מהיר ויציב ,מאחר שהאפליקציה יכולה להתמודד עם אובדן קל של חבילות מבלי לפגוע בחוויית המשתמש.
- Spotify עדיין משתמש במעט TCP, אך עיקר ההעברה מתבצעת באמצעות פרוטוקולי UDP להזרמה מהירה עם השהיה מינימלית.

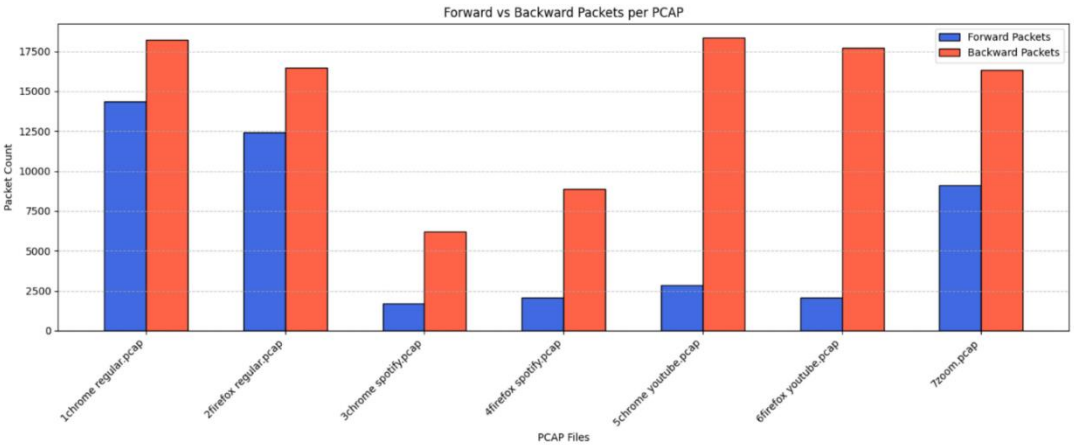
בגלישה רגילה יש רוב מוחלט של TCP,

כי הדפדפנים מבסיס את התקשורת שלהם על פרוטוקולי HTTP/HTTPS אשר דורשים אמינות גבוהה, אישור קבלת חבילות (Acknowledgment), מנגנוני תיקון שגיאות ובקרת זרימה, המובטחים על ידי פרוטוקול TCP.

9. Forward vs Backward Packets per PCAP :

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

- **Forward Packets כחול** – (חבילות שנשלחו מהלקוח (המחשב/דפדפן).
- **Backward Packets כתום** – (חבילות שנשלחו אל הלקוח).
- ההשוואה בין הערכים מסייעת להבין את אופי התקשורת של כל שירות ואת מידת



עומס הנתונים שנשלח מ/אל הלקוח.

מסקנות מהגרף:

<u>בתעבורת דפדפן יחס הזרימה קרוב ל-1, כלומר שליחת וקבלת חבילות נעשית בצורה מאוד מאוזנת</u>

- בגלישה רגילה ,(Chrome/Firefox Regular) הלקוח שולח בקשות HTTP רבות , והשרת משיב בחבילות נתונים המכילות משאבים.
- תעבורת דפדפן נוטה להיות מאוזנת יותר מכיוון שהיא מערבת בקשות מרובות וקבלת נתונים במקטעים קטנים יחסית.

<u>ביוטיוב יחס השליחה קרוב מאוד ל-0, כלומר הלקוח שולח מעט חבילות אך מקבל כמות עצומה של חבילות מהשרת.</u>

- YouTube מבוסס על סטרימינג וידאו ,שבו הלקוח שולח מעט בקשות למשל בקשת GET לסרטון , אך השרת משיב בכמות עצומה של חבילות המכילות את הווידאו עצמו.

- פרוטוקולים כמו QUIC (HTTP3) ו DASH Streaming גורמים לכך שהשרת שולח נפח נתונים גבוה בצורה רציפה ,בעוד שהלקוח כמעט ולא שולח חבילות.
- לכן, יחס ה Forward-ל Backward קרוב מאוד ל-0, כי רוב התעבורה היא בכיוון אחד – אל הלקוח.

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

- Spotify עובד במנגנון buffering בו השיר יורד במקטעים גדולים במקום להזרים נתונים באופן מתמשך כמו YouTube
- הלקוח שולח מעט חבילות ,אך השרת מחזיר כמות גבוהה של חבילות אודיו.
- בניגוד ליוטיוב, בווידאו צריך להמשיך לקבל חבילות כל הזמן, בספוטיפיי יש פרקי זמן של השמעה ללא צורך בנתונים נוספים, ולכן מספר החבילות החוזרות נמוך יותר.
- לכן, יחס השליחה נמוך, אך עדיין גבוה מעט יותר מאשר ביוטיוב.

<u>לגבי ,Zoom קשה לחלץ דפוס קבוע וחד-משמעי של יחס ,Forward/Backward כיוון שיחס זה מושפע ממספר משתנים הקשורים לאיכות השיחה והתנאים ברשת:</u>

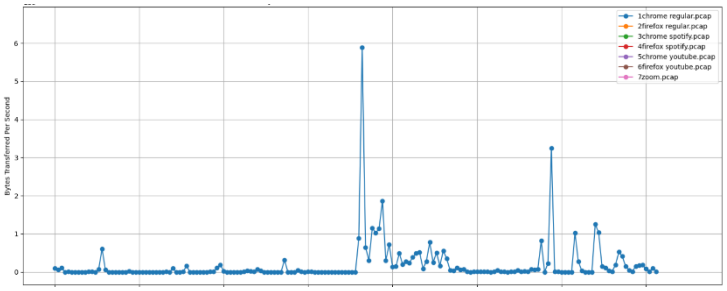
- כמות החבילות הכוללת שנשלחת ב Zoom-תלויה ישירות באיכות החיבור של המשתתפים, ברוחב הפס הזמין, באיכות הווידאו והאודיו (למשל, מצלמה, מיקרופון), וביכולת הדחיסה והקידוד בזמן אמת.
- עם זאת, ניתן להבחין שבאופן כללי Zoom שואף לתקשורת סימטרית יותר) יחס Forward/Backward קרוב יותר ל-1 (ביחס לשירותי סטרימינג חד-כיווניים כמו YouTubeו- Spotify- זאת משום ש- Zoom הוא שירות אינטראקטיבי, שבו כל המשתתפים שולחים ומקבלים נתונים באופן רציף במהלך השיחה.

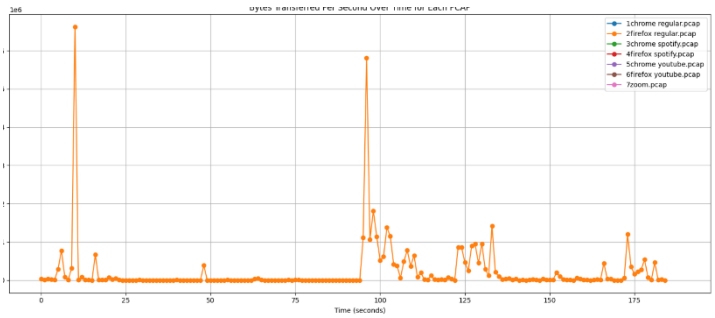
:Flow Volume Per Second.10

גרפים אלו מציגים את כמות הנתונים (בבייטים) שהועברה בכל שנייה לאורך זמן עבור כל תעבורה.

המטרה היא לזהות את מאפייני הדפוס של תעבורה שונה – האם היא אחידה, מתפרצת, יציבה או משתנה בתדירות גבוהה.

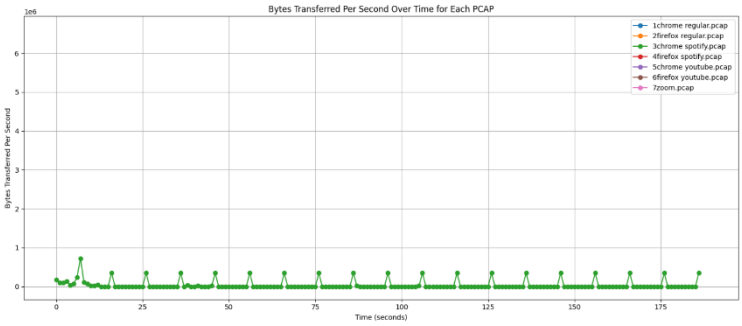
Chrome,Firefox גלישה רגילה:

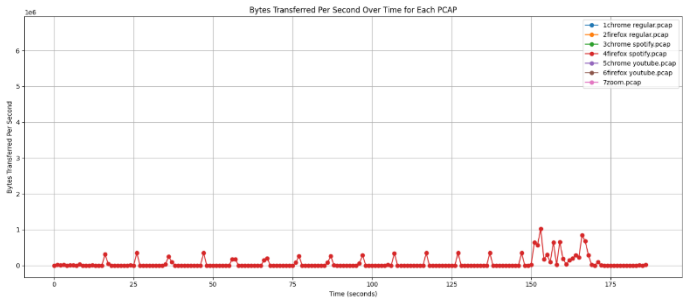




- **מסקנה :**תעבורת הגלישה מציגה קפיצות לא סדירות בכמות הבתים המועברת, עם "שיאים" ברורים ופרקי זמן של כמעט אפס פעילות.
- **סיבה :**בגלישה רגילה, הנתונים נשלחים ונטענים באופן גושי ,(burst) בעיקר כאשר המשתמש נכנס לעמודים חדשים או טוען תוכן. אין רצף של נתונים לאורך כל האינטראקציה כמו בשאר הגלישות בהשוואה שלנו.

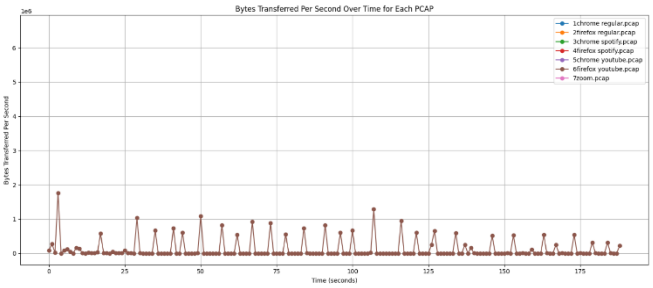
ספוטיפיי:(Chrome + Firefox)

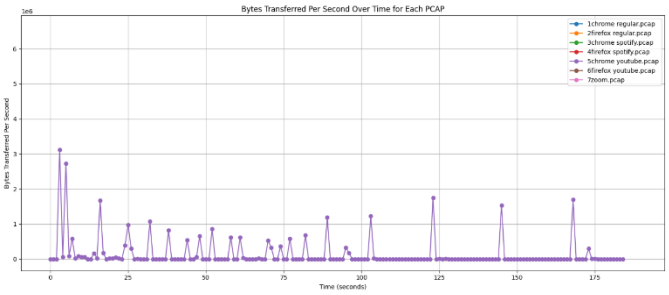




- **מסקנה :**נפח התעבורה של ספוטיפיי מציג תבנית מחזורית – שיאים נמוכים, קבועים, חוזרים כל מספר שניות, בעלי מרווחים קבועים יחסית ביניהם כאשר ברקע התעבורה יציבה מאוד.
- **סיבה :** ספוטיפיי משתמש במנגנון של "buffering" כלומר טוען בלוקים קטנים של מוזיקה מראש באופן קבוע, מה שיוצר תעבורה יציבה יחסית.
המחזוריות משקפת את אופן טעינת קטעי השירים ,ולכן קצב המחזוריות קבוע פחות או יותר בהתאם לאורך המקטעים הנטענים. ומכיוון שזו מוזיקה יש לה BPM קבוע המתבטא בתעבורת הנתונים וברציפותה ואחידותה באופן ציורי.

יוטיוב:(Chrome + Firefox)



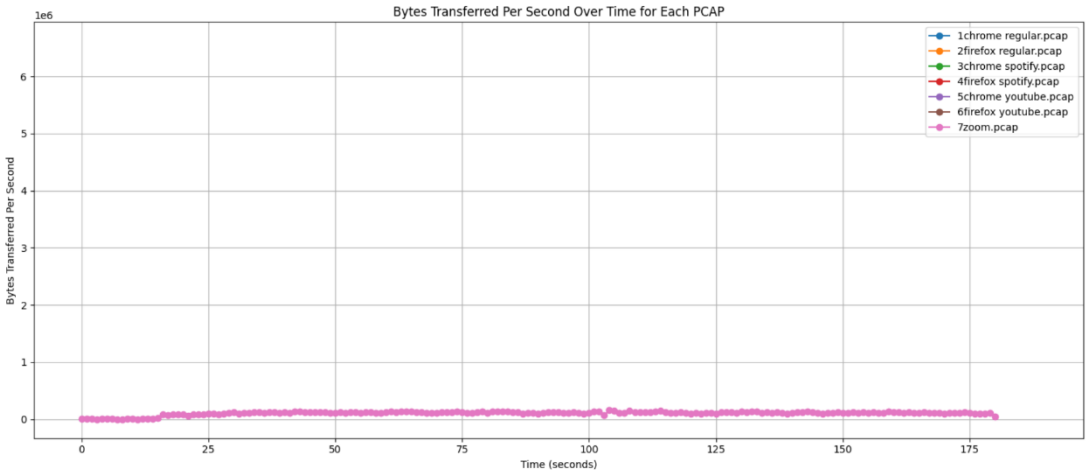


- **מסקנה :**תעבורת יוטיוב מאופיינת בשיאים גבוהים משל ספוטיפיי, בעלי גובה לא קבוע, עם מרווחים לא קבועים, עם פעילות רבה בהתחלה ולאחר מכן ירידה משמעותית בפעילות.

- **סיבה :** יוטיוב מבצע – "pre-buffering" הוא טוען מקטעים גדולים של הווידאו מראש, ואז מפסיק לזמן מה, עד שיש צורך להוריד עוד חלק. זה גורם לתעבורה להיות קפיצית אך חזקה. בנוסף עקב כך שהוא טוען וידאו, אודיו מולו הוא זניח בנפח, ולכן השיאים הגבוהים שלו לעומת זום וספוטיפיי מאוד ברורים.

בנוסף כיוון שהאודיו זניח והתמונה היא דבר משתנה כאשר מדובר בסרטון, הגיוני מאוד שהמרווחים והשיאים יהיו לא קבועים גם כן.

זום:

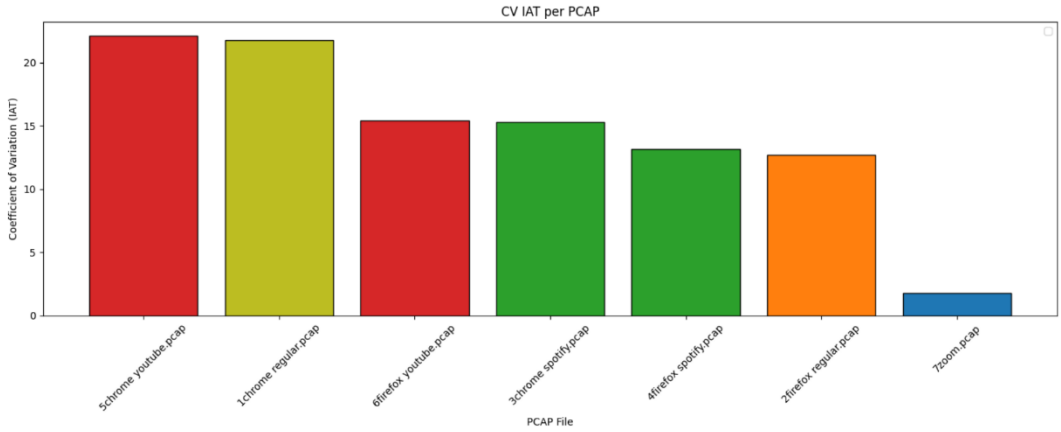


- **מסקנה :**תעבורת זום היא בעלת קצב קבוע יחסית, כמעט ללא תנודות.
- **סיבה :**כיוון שמדובר בשיחת וידאו בזמן אמת (real-time) , זום חייב לשמור על קצב תעבורה אחיד כדי להבטיח חוויית שיחה רציפה. לכן, הוא משדר באופן עקבי ובקצבים קבועים, גם אם כמות הנתונים המועברת בשנייה יחסית נמוכה לעומת יוטיוב למשל. בנוסף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

11.IAT-CV(Burstiness Factor):

כמו שצויין במאמרים CV בהקשר של ניטור רשת מחושב ע"י חלוקת השונות של הזמנים בין החבילות בממוצע של הזמנים בין החבילות.

מהות הגרף היא להציג את מידת ההתפרצות (Burstiness) של זמני ההגעה בין החבילות (Inter-Arrival Times) בקובצי ה PCAP השונים, כפי שנמדד באמצעות מקדם השונות (CV). ככל שהערך גבוה יותר, כך שונות זמני ההגעה בין החבילות גדולה יותר, מה שמעיד



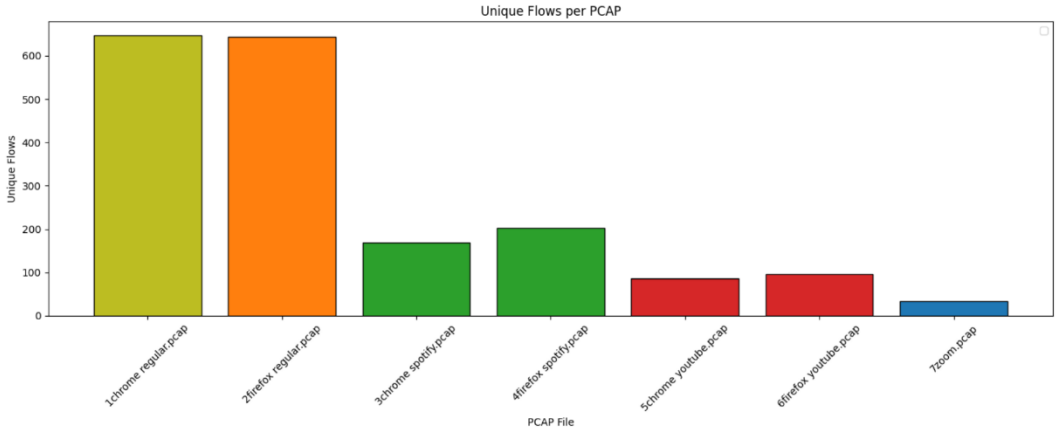
על דפוסי שליחה פחות סדירים.

מסקנות מהגרף:

Zoom מקיים תעבורה הרבה יותר רציפה,יציבה ואחידה (חסרת התפרצויות), זאת כדי לקיים תקשורת רציפה וברורה בין משתתפי השיחה.

:Unique Flows.12

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



מסקנות מהגרף:

לכל סוג תעבורה, כמות הזרימות הייחודיות שבה מעניקה לה טביעת אצבע ייחודית משלה, שכן ניתן להבדיל בין כל הגלישות בגרף זה ביתר קלות.

חלק 4(נקודת מבט של תוקף):

בחלק זה נתבקשנו להקליט שוב תעבורה ולבדוק עד לאיזו רמה נוכל לגלות באילו אפליקציות השתמש הנתקף, כאשר התעבורה מוצפנת, בשתי מקרים:

כאשר יש לנו גישה בכל pcap ל:

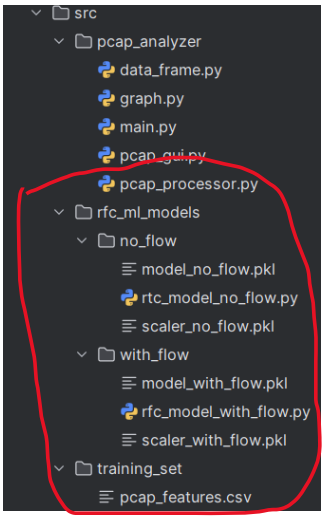
- גדלים, זמני ההגעה וhashi של ה(src ip,ds tip, src port, dst port) של כל חבילה (Flow ID).
- גדלים וזמנים של כל חבילה.

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

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 ישנן 75 הקלטות).
בעזרת ניסוי ותהייה רבים של מאפייני התעבורה בהם נתקלנו כשימושיים לאורך המטלה, הגענו לאלגוריתם שמזהה את התעבורה גם בקבוצת הבדיקה (3 מכל סוג) וגם ב50 שלא השתמשנו באחוזים גבוהים למדי.

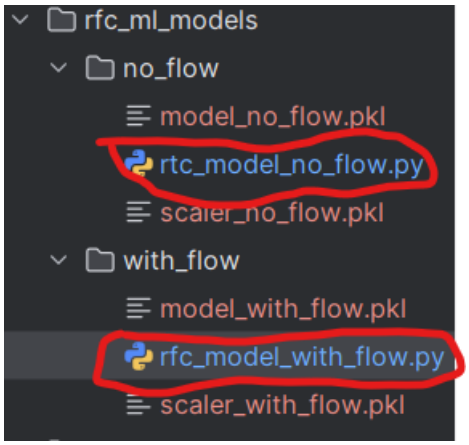
לפני הסבר המימוש נציין שכדי להגיע לאלגוריתם למידת מכונה שמצליח לזהות תעבורה באופן כלל מערכתי/ כלל גיאוגרפי היינו זקוקים לגישה למשאבים רבים ומאגרי מידע של הקלטות בעלי מאות אלפי הקלטות מאזורים גיאוגרפיים שונים, וממערכות שונות, כפי שלמדנו מהמאמרים. אך עבור מימדי המטלה הגענו לתוצאות לא רעות, ואת המסקנות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

מימוש



בחלקו הראשון של הקוד, היה עלינו לממש ניתוח גרפי וויזואלי על הקלטות המייצגות כל סוג תעבורה, אשר הצגנו וניתחנו בחלק 3, כעת נתעסק בקבצים המסומנים בעיגול.

בחלק זה של המטלה על מנת להישאר צמודים להנחיות, בשתי קבצי ה rfc_model המסומנים בתמונה,
הקלט של האלגוריתמים ללמידה, הינו קובץ csv(שתואר מקודם), אשר הופק ע"י ה pcap_processor.py, אשר הוכיח עצמו שימושי בשתי חלקי המטלה (הניתוח הגרפי והחלק של התוקף).



עבור כל אחד מקבצי הpcap שבקבוצת האימון והבדיקה (סהכ 75 קבצי pcapng אשר ניתנים להורדה בקישור בדף הראשון של הPDF), הprocessor מחלץ אך ורק את המידע אשר מותר לתוקף לדעת, לתוך training_features.csv שמשמש בתור ה **Data Set** שלנו.
לכל אחד משתי המקרים בחרנו סט של מאפיינים על פיו הוא ילמד את הדפוסים הקיימים המידע שבקבצי ה pcap.
ביצענו finetuning לסט המאפיינים ולhyper parameters של עומקי העצים שמחליטים באלגוריתם, מצאנו את המאפיינים הבאים כאלה ששימשו את אחוזי הסיווג שלנו ברמה המיטבית:

בלי FlowID

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

]
```

עם FlowID

```
FEATURE_SELECTION = [
    "cv_iat",
    "std_ps",
    # Flow-based features:
    "flow_count",
    # Newly added flow-based pa
    "std_of_flow_pkt_size_std"
]
```

בחרנו במאפיינים אלה אחרי ניסויים רבים של ניסוי ותהייה, אשר הביאו אותנו לתוצאות המיטביות מבחינת סיווג התעבורות, ומטריצת בלבול המבחינה בסיווג תלוי אופי שירות במקרה ראשון ומטריצה המבחינה בסיווג מדויק במקרה השני ככל הניתן.
את התוצאות נציג לאחר הסבר המאפיינים.

עבור כל מאפיין שהיה בשימוש, השתמשנו בMinMax Scale על מנת לנרמל אותו ביחס לשאר החבילות באותו ה pcap, ולאחכ מכן הזנו אותו לRFC.
הנרמול 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 ה pcap.

מאפיין השוואתי	**הסבר**
Std_ps	סטיית התקן של גודל החבילות (Packet Size) . מודדת עד כמה גדלי החבילות משתנים בתוך זרם התעבורה.
Flow_count	מספר הזרמים (Flows) בתוך הקובץ הנמדד. זרם מוגדר לפי הגדרת המטלה.
Std_of_flow_pkt_size_std	סטיית התקן של סטיות התקן של גדלי החבילות בתוך כל זרם. מודד עד כמה פיזור גדלי החבילות משתנה בין זרמים שונים.
Avg_ps	הגודל הממוצע של החבילות, מספק מדד לגבי הגודל הטיפוסי של חבילות בפרק זמן נתון.
Cv_ps	מקדם השונות של גודל החבילות, (השונות חלקי הממוצע של גדלי החבילות).
Avg_iat	הזמן הממוצע בין הגעת חבילה אחת לשנייה. מדד לעומס ולפרקי הזמן בין שליחת נתונים.

Std_iat	סטיית התקן של זמני ההגעה (IAT). מודד עד כמה הפרשי זמני ההגעה של החבילות משתנים לאורך הזמן.
Cv_iat	מקדם השונות של הזמנים בין החבילות, (השונות חלקי הממוצע של הזמנים בין החבילות).
Packets_per_second	מספר החבילות הנשלחות או מתקבלות **בשנייה** .מדד לשיעור התעבורה בפרק זמן נתון.

עבור כל scenario העץ בעזרת המאפיינים הללו מקיים את ההחלטות לסווג כל קובץ עד לעומק מסוים, ולבסוף משקלל את כל האפשרויות לטובה ביותר.

לאחר ניסוי ותהייה רבים וכוונון המאפיינים לדיוק הסיווג הגבוה ביותר אלה התוצאות אשר הניבו את התוצאה במיטבית בשתי המקרים(בעמוד הבא):

תוצאות הניסוי:

בלי FlowID

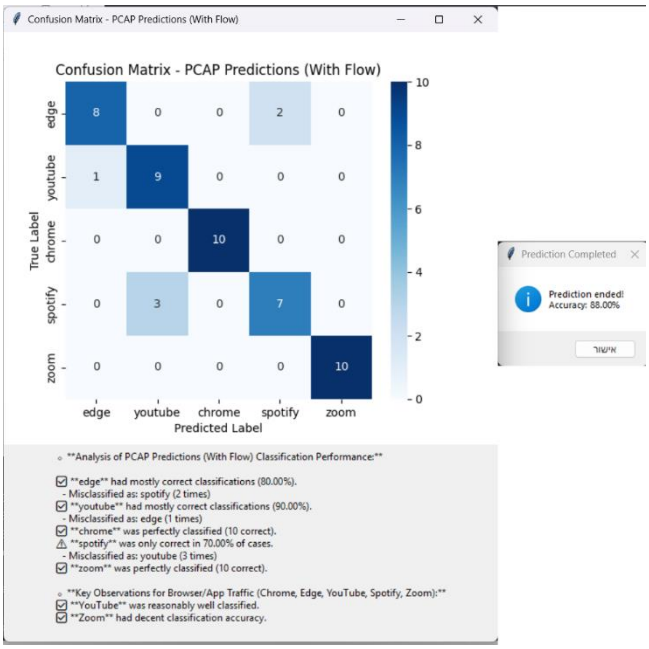
עם FlowID

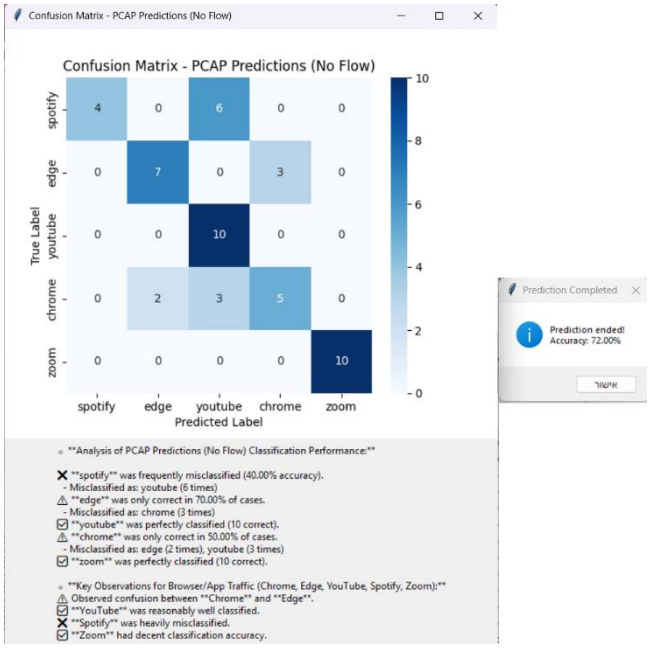
תוצאות אימון האלגוריתם-

```
✅ 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
```

```
✅ 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.
```

וכאשר בחנו אותו בנוסף על 10 הקלטות מכל סוג מהמאגר, אשר לא הוזנו לקוד כחלק מה **Data Set** קיבלנו:





ניתן לראות בבירור שלאחר מציאת שילוב המאפיינים שמביא לאחוז דיוק הסיווג הגבוה ביותר בכל סוג מקרה:

- אחוזי דיוק הסיווג גבוהים משמעותית כאשר נתונים לנו ה Hash של הFlow ID של כל חבילה.
- כאשר לא נתונים לנו הFlow ID לכל חבילה, קשה לסווג את סוג האפליקציה הספציפית אך עדיין ניתן לזהות את אופי השירות.

לדוגמה בתמונה הימנית:

- ספוטיפיי לא סווג נכון,ב 60 אחוז ממספר המופעים שלו, אך עדיין סווג כשירות שמבצע Streaming .
- גם כאשר edge לא סווג נכון הוא סווג כדפדפן (chrome).

רוב טעויות הסיווג נעשו לאותו אופי שירות.

מסקנות הניסוי והמטלה:

- גם כאשר התעבורה מוצפנת, לתוקף יש את הפוטנציאל לזהות את סוג האפליקציה/דפדפן באחוזי סיווג מאוד גבוהים. אם יש לו גישה למאגר מידע בקנה מידה גלובלי של הקלטות, הוא יכול להגיע לאחוזי דיוק בסיווג העולים על 90%!

- **כאשר לתוקף יש גישה אך ורק לגדלי החבילות וזמני ההגעה**, הוא יכול בעזרת חישובים סטטיסטיים, וזיהוי דפוסים, לזהות באותם אחוזי דיוק מהמסקנה לעיל את אופי האפליקציה בה נעשה שימוש (גלישה אקראית/ צפייה בסרטון/ השמעת מוזיקה/ שיחה רציפה ברשת). אך לזיהוי סוג סוג האפליקציה הוא זקוק לטביעת אצבע ייחודית הקשורה אליו באופן ישיר.

- **כאשר לתוקף יש גישה לגדלי החבילות, זמני ההגעה ונתונים סטטיסטיים שטחיים** לגבי הזרימות (כלומר ללא מידע על כתובת השולח/ המקבל ועל התווך בו נעשות השליחות של המידע), הוא יכול לזהות אפילו את סוג האפליקציה באחוזי דיוק מהמסקנה הראשונה.

- הנתונים הסטטיסטיים השטחיים העולים מניתוח הזרימות בתעבורת רשת, מייצגים טביעת אצבע ייחודית של שירותים שונים.

- גם כאשר התעבורה מוצפנת, לתוקף יש את הפוטנציאל לזהות את סוג האפליקציה/דפדפן באחוזי סיווג מאוד גבוהים. אם יש לו גישה למאגר מידע בקנה מידה גלובלי של הקלטות, הוא יכול להגיע לאחוזי דיוק בסיווג העולים על 90%!

- **כאשר לתוקף יש גישה אך ורק לגדלי החבילות וזמני ההגעה**, הוא יכול בעזרת חישובים סטטיסטיים, וזיהוי דפוסים, לזהות באותם אחוזי דיוק מהמסקנה לעיל את אופי האפליקציה בה נעשה שימוש (גלישה אקראית/ צפייה בסרטון/ השמעת מוזיקה/ שיחה רציפה ברשת). אך לזיהוי סוג סוג האפליקציה הוא זקוק לטביעת אצבע ייחודית הקשורה אליו באופן ישיר.

- **כאשר לתוקף יש גישה לגדלי החבילות, זמני ההגעה ונתונים סטטיסטיים שטחיים לגבי הזרימות** (כלומר ללא מידע על כתובת השולח/ המקבל ועל התווך בו נעשות השליחות של המידע), הוא יכול לזהות אפילו את סוג האפליקציה באחוזי דיוק מהמסקנה הראשונה.

- הנתונים הסטטיסטיים השטחיים העולים מניתוח הזרימות בתעבורת רשת, מייצגים טביעת אצבע ייחודית של שירותים שונים (אותם סיכמנו בסוף המטלה).

<u>כיצד נמתן את יכולותיו של התוקף לזהות את התעבורה?</u>

1. Padding (הוספת "רעש" לתעבורה)

התקפה מתבסס על זיהוי דפוסי גודל חבילות וזמני שליחה מדויקים. אחת הדרכים למתן זאת היא Padding (הרחבת חבילות), שמטרתה:

- להפוך את החבילות לאחידות בגודלן או להוסיף גודל רנדומלי, וכך לטשטש את מאפייני התעבורה הייחודיים לכל אפליקציה.

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

2. טשטוש הזמנים (Timing Obfuscation)

- מכיוון שדפוס הזמנים (Inter-Arrival Times) של החבילות מהווה פרמטר לזיהוי האפליקציה, ניתן להוסיף עיכובים או לשלוח חבילות במרווחי זמן קבועים ,(Constant Bitrate) או במרווחים אקראיים.

אסטרטגיה זו מטשטשת את החתימה של אפליקציות כמו Zoom או ,Skype המאופיינות בתעבורה רציפה ואחידה בזמנים.

3. Multiplexing (איחוד זרמי נתונים)

- שימוש בשיטות כמו Multiplexing או חיבור מרובה 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 התוקף לזהות אפליקציה ספציפית מתוך זרם נתונים אחד גדול.

4. הצפנת מידע מזהה (כמו SNI)

- פרוטוקולים כמו TLS 1.3 עם Encrypted ClientHello (ECH) מסתירים מידע שבעבר היה גלוי וחשוף לזיהוי, כגון ה.(SNI) Server Name Indication-

- למרות ש ECH מקשה על סיווג מוקדם ,(Early Classification) עדיין נותר מידע לא מוצפן ברמת ה handshake (כמו שלמדנו במאמר Early Client Hello) לכן, שילוב ECH יחד עם Padding או טשטוש הזמנים יכול להיות יעיל במיוחד.

5. יצירת "רעש" סינטטי (Dummy Packets)

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

6. שימוש בתעבורה סימטרית (Symmetric Traffic)

- הפיכת זרימת התעבורה ליותר סימטרית (שווה פחות או יותר בכמות הנתונים המועברת בשני הכיוונים) 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 אחיד יכול להקשות על זיהוי החתימה של אפליקציות שעושות שימוש בחבילות גדולות וקלות לזיהוי) כמו Youtube או Netflix.

8. שימוש בפרוקסי ובשכבות הצפנה מרובות

- לדוגמה, שימוש ב VPN עם TLS או שילוב Tor ו VPNמעלה 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 גדלים שונים של חבילות ותבניות שליחה, על פי התראות או סימנים ראשוניים של ניסיון זיהוי.

<u>טביעות אצבע ייחודיות לכל סוג אפליקציה מהמטלה:</u>

דפדפן בגלישה אקראית:

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

Youtube (צפייה בסרטונים)

- תעבורה עם חבילות גדולות וממושכות.
- זרימה רציפה לאורך זמן, עם מרווחים קבועים יחסית בין החבילות.
- מאפיין התנהגות של "**Buffering**" פרצי מידע גדולים בתחילת כל סרטון, ואז מעבר לתעבורה קבועה ואיטית יותר.
- יחס **אסימטרי מאוד** של שליחה לעומת קבלה, לטובת קבלה.
- זמני הגעה קבועים יחסית לאורך כל הסרטון לאחר שלב הBuffering.

Spotify (Audio streaming)

- תעבורה בעלת חבילות בינוניות-גדולות (פחות מווידאו, אך גדולות יותר מגלישה רגילה).
- זרימה יציבה יחסית ,קבועה וממושכת ,עם התנהגות רציפה לאורך ההשמעה.
- זמני הגעה קבועים ואחידים יחסית, יציבות לאורך ההאזנה.
- יחס **אסימטרי ברור** בעיקר קבלה מעט מאוד שליחה.
- כמות החבילות לכל פרק זמן נתון אחידה למדי לאחר התחלה קצרה של טעינה.

Zoom (Video Conferencing)

- תעבורה רציפה של חבילות קטנות-בינוניות בתדירות גבוהה.

- זמני הגעה קצרים וקבועים רוב הזמן מדובר בתקשורת זמן-אמת ללא השהיה.
- יחס תעבורה סימטרי באופן יחסי (כמות גבוהה של שליחה וקבלה במקביל).
- מאפייני החבילות יציבים בגודלן (קטנות יחסית לעומת סטרימינג וידאו כמו YouTube)
- זרימה דו-כיוונית ויציבה לכל אורך השיחה, ללא הפסקות משמעותיות.