Communication Networks -

Ilya Bagaev, Yonatan Gabay, Hadas Raviv, Or Deri

Table Of Contents -

PART I - Questions And Answers

שאלה 1: משתמש מדווח שהעברת הקבצים שלו איטית, ועליך לנתח את שכבת התעבורה כדי לזהות את הסיבות האפשריות לכך. מהם הגורמים שעשויים להשפיע על ההעברה וכיצד היינו פותרים אותה?

פתרון:

הגורמים המרכזיים שעשויים להשפיע על ההעברה הם:

1. עומס ברשת- עומס ברשת מתרחש כאשר יותר מדי מקורות שולחים נתונים בקצב גבוה שהרשת אינה מסוגלת להתמודד איתו, מה שמוביל לעיכובים, איבוד פאקטות, ופגיעה בביצועי התקשורת. עומס ברשת נגרם ממגוון סיבות

-עיכובים בתור של הנתב: כאשר חבילות נתונים מחכות בנתב להישלח, מה שמוביל לעיכוב השליחה והנתב עלול לזרוק חבילות כדי להפחית מהעומס מה שיוביל לאיבוד חבילות ויגרור שליחה מחדש. וסה"כ עיכוב משמעותי בזמן ההעברה

-גודל הבאפר: כאשר הבאפר מתמלא חבילות עלולות להיזרק וללכת לאיבוד- מה שיוביל לשליחה מחודשת של החבילות, מה שמאט את קצב העברת הנתונים.

2. RTT- מהירות הזמן שחבילה מגיעה ליעדה וחוזרת, הוא גורם מושפע מהרשת שמושפע מכמה דברים בעיקרם שמשפיעים על מהירות החבילה:

זמן התפשטות- המושפע מאורך הקו וממהירות התפשטות בקווים. ככל שאורך הקו גדול וזמן התפשטות הקווים קטן כך הזמן הכולל יהיה ארוך יותר.

זמן שידור- המושפע מגודל המידע וקצב השידור. ככל שגודל המידע גדול קצב השידור קטן הזמן הכולל גדל.

3. רוחב פס נמוך - רוחב פס נמוך מתרחש כאשר קצב העברת הנתונים נמוך ביחס לכמות המידע שצריך להעביר, מה שמוביל לעיכובים והאטה בביצועים.

איך נפתור את הבעיה? על ידי שימוש בבקרת עומסים. למשל שימוש במנגנוני בקרת עומסים:

חלון שליחה – sliding window: ב-TCP, גודל חלון השליחה קובע כמה נתונים השולח יכול לשלוח לפני קבלת אישור (ACK) מהמקבל. ברשת מהירה עם RTT קצר, הגדלת חלון השליחה תאפשר ניצול טוב יותר של רוחב הפס ותאיץ את ההעברה. ברשת עמוסה או עם רוחב פס נמוך, TCP יקטין את גודל החלון כדי למנוע איבוד נתונים ועומסים נוספים.

שליחה מבוקרת - אלגוריתם AIMD – עליה איטית בשליחת החבילות והרדה דרסטית בהינתן עומס .

שאלה 2: נתח את ההשפעות של מנגנון בקרת זרימה בTCP על העברת נתונים. איך זה ישפיע על ההעברת נתונים כשאר לשולח יש כוח עיבוד גבוה בהרבה מהמקבל?

פתרון:

בקרת הזרימה של : TCP נועדה למנוע מצב שבו השולח מעביר נתונים מהר יותר ממה שהמקבל מסוגל לעבד, מה שעלול לגרום לעומס, איבוד חבילות וירידה בביצועים.

TCP פותר את הבעיה הזו באמצעות שדה ה- rwnd (Receiver Window), שמופיע בכותרת ה-TCP ומציין כמה נתונים המקבל יכול לקבל בכל רגע נתון. מנגנון זה מאפשר לשולח להתאים את קצב שליחת הנתונים בהתאם ליכולת העיבוד של המקבל, כך שלא יציף אותו בכמות מידע שהוא אינו מסוגל להתמודד איתה. כאשר הבאפר של המקבל מתמלא ואין לו עוד מקום לקבל נתונים, הוא מעדכן את השולח שגודל החלון (rwnd) הוא 0, מה שמונע מהשולח לשלוח חבילות נוספות עד שהמצב יתפנה. יש 2 מצבים לTCP:

TCP Persist: כאשר השולח מקבל מהמקבל מצב של חלון 0 אז הוא מחכה לעדכון נוסף מהמקבל כאשר גודל החלון יעלה. אם העדכון יאבד- השולח ימשיך לחשוב שגודל החלון הוא 0.

TCP Persist state: בשביל להתגבר על איבוד העדכון השולח מדיי פעם ישלח חבילות ממש קטנות של בייט בודד והמקבל יענה לו באישור קבלה – ACK ויעדכן את מצב הrwnd שלו.

ובעצם השפעות של מנגנון הזרימה על העברת נתונים הם שבקרת הזרימה של TCP מווסתת את קצב שליחת הנתונים בהתאם ליכולת המקבל לעבד אותם. כלומר שינוי דינאמי של העברת הנתונים בהתאמה ליכולת המקבל.

כאשר לשולח יש כוח עיבוד גבוה משמעותית מהמקבל, נוצר חוסר איזון בקצב עיבוד הנתונים, מה שמוביל להאטה משמעותית בתקשורת. המקבל מתקשה לעבד את החבילות במהירות שבה הן נשלחות, מה שגורם לחלון הקבלה (rwnd) שלו להצטמצם בהדרגה עד לאפס. במצב זה, השולח נאלץ לעצור את שליחת הנתונים ולהמתין עד שהמקבל יתפנה ויעדכן את החלון מחדש. התוצאה היא השהיות ממושכות וחזרות תכופות למצבי חלון אפס, שבמהלכן השולח אינו מנצל את מלוא יכולותיו ואינו מצליח לשמור על קצב שידור גבוה. כתוצאה מכך, רוחב הפס אינו מנוצל כראוי, והשולח מבזבז את כוח העיבוד הגבוה שלו בהמתנה לאישורים מהמקבל, במקום להעביר נתונים בקצב אופטימלי.

שאלה 3: נתח את תפקיד הניתוב ברשת שבה קיימים מספר נתיבים בין המקור ליעד. כיצד בחירת הנתיב משפיעה על ביצועי הרשת, ואילו גורמים יש לקחת בחשבון בהחלטות ניתוב?

פתרון:

תפקיד הניתוב: ברשת בה קיימים מספר נתבים בין המקור ליעד, תפקיד הניתוב הוא לאסוף נתונים על המתרחש בדרך ולהחליט מהי הדרך היעילה ביותר להעברת הנתונים. כמו כן, תפקיד הניתוב הוא גם לאזן עומסים- פיצול התעבורה למספר נתיבים במקביל כדי לנצל את רוחב הפס ולמנוע "סתימה" של מסלול אחד בעוד האחרים נותרים ריקים יחסית. היכולת הזו משפרת את הקצב ואת היציבות של החיבור הכולל. בכל מקרה, הרשת מתעדכנת באופן דינמי בהתאם לשינויים כמו תקלות מקומיות או תעבורה מוגברת, תוך חישוב מחדש של נתיבים אופטימליים.

בחירת הנתיב ברשת משפיעה ישירות על הביצועים בכך שהיא קובעת את מהירות ואמינות העברת הנתונים. נתיב יעיל מאפשר השהיה נמוכה, ניצול מקסימלי של רוחב הפס ומינימום אובדן חבילות, מה שמשפר את חוויית המשתמש בתקשורת בזמן אמת ובהורדות. לעומת זאת, בחירה לא

אופטימלית עלולה לגרום לעומסים, האטת קצב התעבורה, חוסר יציבות ותקלות בתקשורת

הגורמים שיש לקחת בהחלטת הניתוב הם:

אלגוריתם הניתוב – בהחלטת הניתוב נצטרך לבחור לפי איזה אלגוריתם נלך. דיאגסטרה, בלמן פורד..

רוחב פס והשהיה – נתיב עם רוחב פס גבוה והשהיה נמוכה עדיף לביצועים מהירים יותר.

עומס ברשת – אם נתיב מסוים עמוס, הנתבים עשויים לבחור נתיב חלופי.

אמינות וזמינות – ניתוב יכול להעדיף מסלולים יציבים עם פחות נפילות או אובדן חבילות.

גיבוי ואיזון עומסים – תכנון הנתיבים כך שאם נתיב אחד נופל, יש חלופה, או פיצול עומסים בין כמה מסלולים.

שאלה 4: כיצד MPTCP משפר את ביצועי הרשת?

פתרון:

MPTCPהוא הרחבה של פרוטוקול TCP המאפשרת שימוש במספר נתיבים בו-זמנית עבור אותו חיבור רשת. במקום להעביר את כל הנתונים על גבי נתיב אחד בלבד, כמו ב-TCP רגיל, MPTCP מחלק את התעבורה לכמה תת-חיבורים העוברים דרך מסלולים שונים.

פרוטוקול זה משפר את ביצועי הרשת בכמה היבטים:

שיפור מהירות ההורדה והעלאה – במקום להשתמש רק בנתיב אחד, MPTCP מפצל את הנתונים לכמה נתיבים במקביל, מה שמנצל טוב יותר את רוחב הפס ומאיץ את העברת הנתונים.

גיבוי לחיבורים – אם אחד החיבורים נופל הנתונים ימשיכו לעבור דרך חיבור אחר בלי שהחיבור עצמו יתנתק, מה שחוסך זמן ומונע הפסקות.

הפחתת השהיות– הנתונים מחולקים לכמה מסלולים, ואפשר לבחור בכל רגע את הנתיב הכי מהיר, מה שמקטין את ההשהיה הכללית של התקשורת.

שיפור החוויה של המשתמש– החיבור לרשת נשאר רציף גם אם יש שינויים פתאומיים כך שאין צורך לחכות לחיבור מחדש.

שאלה 5: אתה עוקב אחר תעבורת הרשת ומבחין באובדן חבילות גבוה בין שני נתבים. נתח את הסיבות הפוטנציאליות לאובדן חבילות בשכבת הרשת ובשכבת התעבורה המליץ על צעדים לפתרון הבעיה

פתרון:

הסיבות הפוטנציאליות לאובדן חבילות בשכבת הרשת הן:

1. עומס בנתבים- יכול להיות שקיים עומס גדול בין שני הנתבים ויכולת עיבוד נמוכה של חבילות שגורמת לנתבים לזרוק הרבה חבילות כדי להקל על העומס.
2. בעיות ניתוב– טבלאות ניתוב שגויות או מסלולים לא יציבים עלולים לגרום לחבילות ללכת לאיבוד.
3. TTL פג – אם חבילה מסתובבת זמן רב מדי ברשת בלי להגיע ליעדה, היא תימחק כאשר ערך ה-TTL שלה מגיע לאפס.

הסיבות הפוטנציאליות לאובדן חבילות בשכבת התעבורה הן:

1. גודל חלון TCP לא מותאם – גודל החלון קובע כמה נתונים אפשר לשלוח לפני שמחכים לאישור. אם הוא מוגדר גדול מדי לרשת עם השהיות גבוהות, החבילות יכולות להיאבד כי הקצה השני לא מספיק לאשר אותן בזמן. מצד שני, אם הוא קטן מדי, הניצול של רוחב הפס לא יהיה מיטבי.
2. Timeout לא מותאם נכון – TCP מצפה לקבל אישור (ACK) על כל חבילה שהוא שולח. אם ה-timeout קצר מדי, TCP ישדר מחדש חבילות מוקדם מדי ויגרום לעומס מיותר. אם ה-timeout ארוך מדי, ההתאוששות מאובדן חבילות תהיה איטית מדי.
3. סדר הגעת חבילות - במקרים שבהם חבילות מגיעות לא בסדר המקורי (למשל, נתיבים שונים ברשת), השולח עשוי לפרש זאת כאובדן חבילה ולשדר אותה מחדש למרות שהיא לא באמת אבדה. זה מייצר עומס מיותר.

המלצה לפתרון הבעיות:

1. הורדת עומס מהנתבים- לבדוק אם הנתבים לא חנוקים מתעבורה ולפזר עומסים אם צריך. להגדיר כך שחבילות חשובות יקבלו עדיפות ולא ייזרקו בגלל עומס.
2. בדיקת ניתוב ו-TTL -לעבור על טבלאות הניתוב ולוודא שאין שם טעויות שגורמות לחבילות ללכת לאיבוד. לבדוק אם החבילות מתעכבות או נתקעות בלולאות ברשת ולתקן מסלולים בעייתיים. אם TTL נגמר מהר מדי, אולי צריך לכוון אותו נכון או לשפר את הנתיבים כך שהחבילות יגיעו מהר יותר ליעד.
3. התאמת פרמטרי TCP - לוודא שגודל החלון מותאם לרוחב הפס והשהיות, כדי שלא יהיה גדול מדי (יגרום לאיבוד חבילות) או קטן מדי (יעכב את התקשורת). לכוון נכון את ה-Timeout כדי לא לשדר חבילות מחדש מוקדם מדי (מה שיוצר עומס) או מאוחר מדי (מה שיאט את ההתאוששות).
4. שימוש בתעדוף חבילות – QoS

Part II - Article Reading

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

רקע:

בעידן של הצפנה מתקדמת כמו TLS 1.3 ו-Encrypted ClientHello (ECH), סיווג תעבורה מוקדם (ETC) הופך למאתגר עבור ניהול איכות שירות (QoS) ברשתות. המאמר מציע גישה היברידית חדשה (hRFTC), המשלבת מאפייני TLS בלתי מוצפנים עם נתונים סטטיסטיים של זרמי תעבורה, ומשיגה דיוק של 94.6% בסיווג תעבורה מוצפנת. המחקר כולל מאגר נתונים רחב היקף ומוכיח כי שיטות מבוססות למידת מכונה יכולות לזהות תעבורה גם ללא מידע קריטי. מסקנותיו מדגישות את הצורך במודלים גמישים המותאמים לאזורים גיאוגרפיים שונים.

התרומה העיקרית של המאמר:

בעקבות הקושי לסיווג תעבורה מוקדמת אחרי הצפנה מתקדמת (TLS 1.3 וECH – המפורטים ברקע) התרומה המרכזית של המאמר מתמקדת בפיתוח אלגוריתם חדשני ויעיל, הקרוי (Hybrid Random Forest Traffic Classifier) hRFTC, אשר נועד להתמודד עם הצפנה מתקדמת ולספק סיווג מוקדם של תעבורה מוצפנת.

האלגוריתם נבנה מתוך ההבנה שרוב שיטות הסיווג המסורתיות נסמכו בעבר על זיהוי שרת באמצעות -SNI Server Name Indication גלוי, אך בהתבסס על ECH, מידע קריטי זה מוסתר ואינו זמין עוד למנגנוני הסיווג. כדי להתגבר על המחסום הזה, ה-hRFTC משלב בין שתי משפחות נתונים משלימות: (1) מאפייני TLS שעדיין נותרים גלויים או נחשפים בחלקם –פרטי גרסאות, מיקום וסדר הרחבות וכמו כן ה- Key

Share הדרוש להקמת ההצפנה.
(2) תכונות סטטיסטיות-התנהגותיות של התעבורה כמו התפלגות גדלי החבילות כמות הגדלים הייחודיים, זמני ההגעה בין חבילות, וגם מספר החבילות הנבחנות עד להגעת הנתונים הראשונים מהאפליקציה.
השילוב של המאפיינים האלו הוא שילוב היברידי המנצל את המידע המועט שנותר בפרוטוקול עצמו לצד מדדים סטטיסטיים, ובכך מצליח לייצר הפרדה בין סוגי שירות שונים (למשל וידאו ארוך, וידאו קצר, אודיו, לייב סטרימינג, גלישה כללית ועוד)

בזכות הגישה הזו, המאמר מפגין רמת דיוק מרשימה של עד 94.6%, ומראה כיצד hRFTC גובר משמעותית על אלגוריתמים מבוססי מטא-נתונים בלבד או מבוססי זרם בלבד, המתקשים בתרחיש בו ECH מסתיר את SNI. השילוב ההיברידי של הנתונים והקמת האלגוריתם הזה יצר פתרון זמין ויישומי לאתגרים העכשוויים של סיווג מוקדם ברשתות מודרניות מוצפנות.

אילו מאפייני תעבורה המאמר משתמש בהם ואילו מהם חידושים?

המאמר משתמש במאפייני התעבורה הבאים:

1. מאפיינים מבוססי חבילה – TLS: מאפייני חבילות הם בעצם ניתוח המבנה של כותרות TLS והחלקים הגלויים בחבילה (למשל ב-ClientHello או ב-ServerHello), עוד לפני שההצפנה השיטתית של התוכן נכנסת לפעולה. הרעיון הוא להסתכל על שדות כמו גרסת TLS, Key Share, סדר ההרחבות או כל אינדיקציה אחרת שנשארת

גלויה באופן חלקי. בצורה זו אפשר לפענח סוג של "טביעת אצבע" לאפליקציה או לשירות שמשתמשים בקומבינציה מסוימת של הרחבות, או בגודל נתונים קבוע בכל התחלה של חיבור. למעשה, גישה מבוססת חבילות מתמקדת במה שאפשר ללמוד מהחבילות הראשוניות עצמן.

2. מאפיינים מבוססים זרם: מאפייני זרם הם מאפיינים המבוססים על הניתוח הסטטיסטי של תעבורה לאורך זמן, מבלי להתעמק בתוכן החבילות עצמן. בגישה זו, עוקבים אחר הגדלים של החבילות והמרווחים בינהם למשל ממוצע, חציון, ערכי קיצון וכמות גדלים ייחודיים. לרוב בודקים אותם בנפרד עבור כיוון העלאה וכיוון הורדה , שכן יישומים שונים מבצעים תקשורת מבוססת גודל-חבילה וקצב התעבורה ייחודי.

שני מאפיינים אלו כשלעצמם אינם חידושים, שכן בעבר כן השתמשו בעצם לסיווג התעבורה. החידוש העיקרי הוא מה שאחרי הצפנת TLS 1.3 ו- ETC התשמשו במאפיין תעבורה היברידי ששלב את שתי השיטות, והוא מהווה החידוש הגדול:

אפיון היברידי: השילוב בין מאפייני חבילות ומאפייני זרם מהווה חידוש המאמר, שכן הם יוצרים חיבור ישיר המאפיינים כך שאלגוריתם הסיווג לא מסתמך על מקור מידע

יחיד אלא מנצל באופן אופטימלי את כל הנתונים הגלויים שעדיין זמינים גם אחרי הצפנת ECH.

הרעיון המרכזי בגישה ההיברידית הוא שכל אחת מהשיטות בנפרד, מאפייני הזרם ומאפייני החבילות אינם מספיקים לבד אחרי שהוצגה ההצפנה החדשה ב-TLS 1.3. מצד אחד, מאפייני החבילות נותנים מידע חיוני מהכותרות הגלויות של TLS, אך לאחר הצפנת ClientHello כמות המידע שניתן להפיק ישירות מהחבילות מצטמצמת. מצד שני, מאפייני הזרם (כגון גדלי חבילות, זמני הגעה בין חבילות והתפלגותם) מספקים רמזים על אופי השירות, אך ללא מידע נוסף על השרת והיישום. מאפיינים אלה לבדם לא תמיד מאפשרים הפרדה מדויקת. לכן, המאמר מציע גישה שבה שני סוגי המאפיינים משולבים יחד בתוך וקטור תכונות משותף, כך שהם מזינים מידע זה לזה ומפצים על החסרונות אחד של השני.

כדי לבצע את החיבור הזה בפועל, החוקרים יוצרים וקטור נתונים מובנה שבו כל מאפיין TLS ששרד את ההצפנה מקבל ייצוג מסודר ואחיד. במקביל, הם אוספים נתונים סטטיסטיים על התעבורה עד לנקודה שבה מתקבלת חבילת ה- Application Data הראשונה, מה שמאפשר לסווג את החיבור בשלב מוקדם מאוד (Early

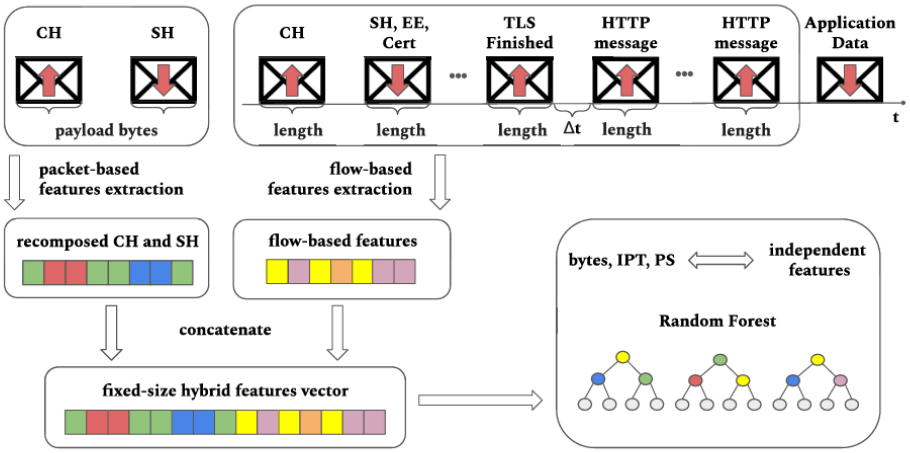


FIGURE 2. Hybrid random forest traffic classifier.

Classification) מבלי להמתין לניתוח ארוך של הזרם כולו. השימוש בווקטור אחיד מבטיח שהאלגוריתם (hRFTC, מבוסס Random Fores) יקבל תמונה מלאה ככל האפשר של החיבור, גם אם חלקים מסוימים של המידע מוצפנים. במאמר נוצג תרשים ויזואלי לאיך חיבור המידע לוקטורים ובסוף לעץ החלטות של סיווג המידע מתבצע:

כמו כן, המאמר מציג התאמה של השיטה ההיברידית גם לפרוטוקול QUIC, המשמש כבסיס ל-HTTP/3, שבו ההצפנה מתחילה מוקדם יותר בהשוואה ל-TLS מבוסס TCP. מאחר שב-QUIC רוב ה-Handshake מוצפן, לא ניתן להסתמך על מאפייני TLS גלויים כפי שנעשה ב-TLS רגיל. כדי להתמודד עם האתגר הזה, החוקרים

מנצלים מאפיינים ייחודיים של QUIC, כגון Connection ID ושינויים בגודל החבילות לצד ניתוח סטטיסטי של זרם התעבורה ובכך, המודל ההיברידי מצליח לסווג גם תעבורה מוצפנת מבוססת QUIC, ומאפשר המשך זיהוי תעבורה מדויק גם בעולם שבו HTTP/3 הופך לדומיננטי, תוך שימוש באותם עקרונות של שילוב בין נתוני מבוססי חבילות ומאפייני זרם.

שזה אכן חידוש כי בעבר הסיווג היה רק לתעבורה מסוג TCP over TCP

והאלגוריתם מצליח להתמודד עם TCP over UDP- שזה בעצם QUIC:

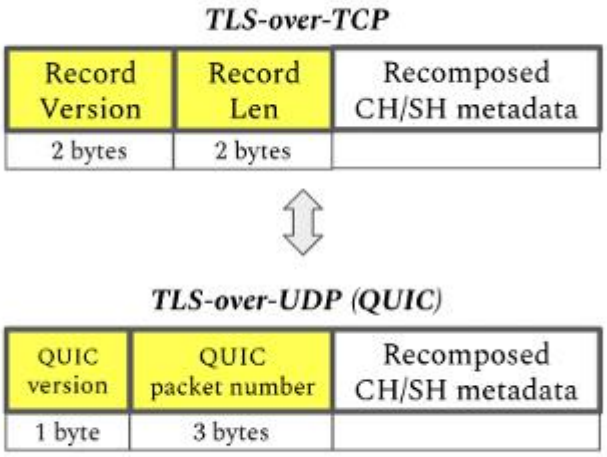
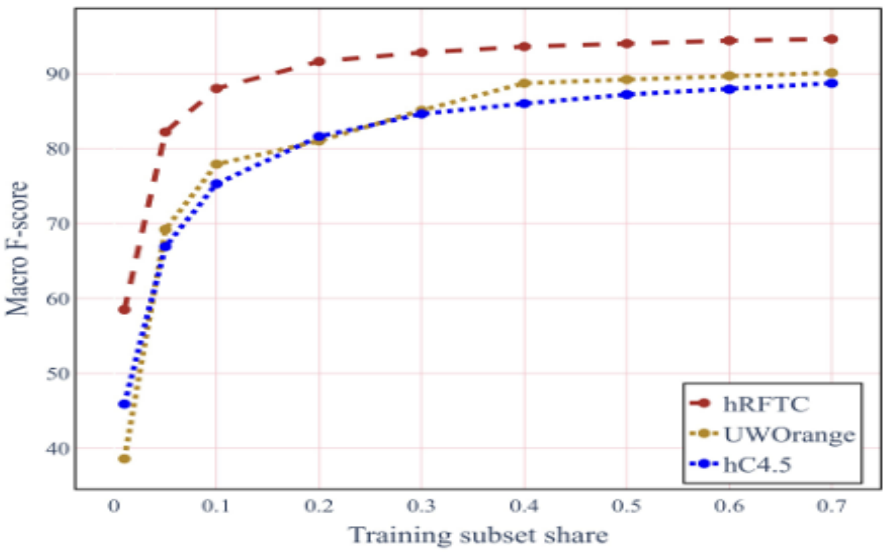


FIGURE 3. Recomposed payload modification for QUIC flows.

<u>מהם התוצאות העיקריות ומהם התובנות העיקריות מהתוצאות?</u>

שהתוצאות העיקריות שמציג המאמר הן שהאלגוריתם ההיברידי (hRFTC) שהוצע מגיע לרמות דיוק גבוהות מאוד בסיווג תעבורה בתרחיש שבו Server Name (SNI) F-Scor (מאקר ריתמים קודמינ , שצונחים לא פעו גרף המצור



שהאלגוריתם hRFTC מגיע מאוד מהר לאחוזי הצלחה מדהימים.

התובנות העיקריות שעולות מהתוצאות האלו הן:

1. חשיבות השילוב ההיברידי - בעוד שגישה המתמקדת אך ורק באחד מהמאפיינים, מאפייני חבילות או מאפייני הזרם תגיע בעידן של ההצפנה היום לאחוזי הצלחה נמוכים מאוד, השילוב בין השתיים מקבל אחוזי הצלחה מרשימים וגבוהים.

2. חשיבות סיווג מוקדם יעיל - החוקרים מראים שבאמצעות עצירת איסוף הנתונים בשלב שבו חבילת Application Data ראשונה מגיעה מהשרת, אפשר לזהות את השירות כבר בתחילת החיבור. המשמעות היא שניתן ליישם את הסיווג בזמן אמת מבלי להמתין לאיסוף מאות חבילות.

3. התאמה לפרוטוקולים חדשים- האלגוריתם הותאם גם לתרחיש QUIC (HTTP/3), מה שמעיד על גמישות המודל להתמודד עם מגמת ההצפנה הגוברת באינטרנט. היכולת לטפל ב-QUIC, שבו ה-Handshake מוצפן מהר יותר, מרחיבה את תחולת השיטה ומבססת אותה כפתרון עדכני גם לעתיד.

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

רקע - מאמר זה מביע טענה חשובה וממוקדת עבור הקורא שמידע שעובר בפרוטוקול HTTPS עשוי להיות חשוף בפני תוקפים. כלומר העובדה שכיום, רוב התעבורה באינטרנט היא אכן מוצפנת ותקיפות פסיביות בעולם התעבורה הן אכן מאתגרות. לעומת זאת, במאמר זה מציגים בפנינו טענה שבה תוקף חיצוני אכן יכול לזהות את מערכת ההפעלה של המשתמש, את הדפדפן והאפליקציה של התעבורה HTTP המוצפנת (HTTPS) וזאת באמצעות האזנה פסיבית לתעבורה של משתמש, אף על פי שהתעבורה מוצפנת בHTTPS/TLS -, המאמר מספק הוכחה לכך שעדיין ניתן לחלץ מידע חשוב על משתמש על סמך דפוסי התעבורה המוצפנת.

התרומה העיקרית של מאמר זה מתבטאת באופנים הבאים:

בחלק Introduction - I, המחברים מציגים בפני הקורא, שלראשונה ניתן לזהות באמצעות ניתוח דפוסי תעבורה את מערכת ההפעלה, הדפדפן והיישום שמשתמש מסוים נמצא בו, כלומר זה נעשה באופן פסיבי ולא מתרחש פה סוג של פעולה אקטיבית של התוקף כמו למשל קריאה של תוכן ההצפנה וכו'..

בנוסף לכך, המחברים מציגים בפנינו הוכחה לזיהוי Tuple חדשני , אשר ניתן בהחלט לסווג בצורה מדויקת את <OS,Browser,Application> מתוך נתוני התעבורה ללא צורך בפענוח המידע. ישנו הצעה של מאפיינים חדשים של ניתוח תעבורה לצד המאפיינים הבסיסיים.

נראה בהמשך שאכן מאפיינים חדשניים אלה - שמתבססים על מאפייני הTCL , -SSL ודפוס התנהגות מתפרץ (Bursty) בהתנהגות של הדפדפן אשר יביאו לשיפור מהותי בדיוק של הסיווג.

נסיים את סעיף זה בעוד תרומה שאנו חושבים שהיא די משמעותית. המאמר מציג בפנינו Data-set עם מעל ל-20000 דגימות, דגימות אלה בוצעו לפי המאמר בעזרת שיטות למידת מכונה. מחברי המאמר השתמשו ב-SVM (Support Vector Machine) ובנוסף ב-(RBF (Radial Basis Function, אשר הביאו לדיוק של 96.06% לעומת השיטה הנאיבית שהביאה רק ל- 32.34%.

<u>באילו מאפייני תעבורה המאמר משתמש, אילו מאפיינים הם חדשניים?</u>

במאמר מוצג בפנינו באופן ברור ומוחלט את כלל מאפייני התעבורה שנעשה בהם שימוש, וביניהם גם המאפיינים החדשניים. נצרף צילומי מסך ונסביר עליהם -

לפי צילומי המסך מהמאמר, ניתן לראות באופן ברור באילו מאפיינים המחברים השתמשו לצורך ניתוח התעבורה. מאפיינים אלה הוגדרו לפי שני חלקים.

Forward packets
Forward total Bytes
Min forward inter arrival time difference
Max forward inter arrival time difference
Mean forward inter arrival time difference
STD forward inter arrival time difference
Mean forward packets
STD forward packets
Backward packets
Backward total Bytes
Min backward inter arrival time difference
Max backward inter arrival time difference
Mean backward inter arrival time difference
STD backward inter arrival time difference
Mean backward packets
STD backward packets
Mean forward TTL value
Minimum forward packet
Minimum backward packet
Maximum forward packet
Maximum backward packet
Total packets
Minimum packet size
Maximum packet size
Mean packet size
Packet size variance

(a) base features

TCP initial window size
TCP window scaling factor
SSL compression methods
SSL extension count
SSL chiper methods
SSL session ID len
Forward peak MAX throughput
Mean throughput of backward peaks
Max throughput of backward peaks
Backward min peak throughput
Backward STD peak throughput
Forward number of bursts
Backward number of bursts
Forward min peak throughput
Mean throughput of forward peaks
Forward STD peak throughput
Mean backward peak inter arrival time diff
Minimum backward peak inter arrival time diff
Maximum backward peak inter arrival time diff
STD backward peak inter arrival time diff
Mean forward peak inter arrival time diff
Minimum forward peak inter arrival time diff
Maximum forward peak inter arrival time diff
STD forward peak inter arrival time diff
Keep alive packets
TCP Maxiumu Segment Size
Forward SSL Version

(b) new features

TABLE I: The two sets of features used in this paper. The base features are features that " are used in many traffic classification methods. The new features are proposed in this paper." (ציטוט של שני הטבלאות).

צילום מסך של פסקה מהמאמר אשר מסביר את אופן חילוץ המאפיינים -

B. Feature Extraction

This section describes how we extract features from a session and the feature characteristics. Encrypted traffic generally relies on SSL/TLS for secure communication. These protocols are built on top of the TCP/IP suite. The TCP layer receives encrypted data from the above layer and divides data into chunks if the packets exceed the Maximum Segment Size (MSS). Then, for each chunk it adds a TCP header creating a TCP segment. Each TCP segment is encapsulated into an Internet Protocol (IP) datagram. As TCP packets do not include a session identifier, we identify a session using the tuple <Protocol, IP source, IP destination, Port source, Port destination>.

A session contains two flows: forward and backward. A flow is defined as time ordered sequence of TCP packets during a single TCP session. The forward flow is defined as a time series bytes transported by incoming packets only, while the backward flow is defined as a time series bytes transported by outgoing packets only. We use the forward, backward and their combination as a representation of a connection. Additionally we also use time series features such as inter arrival time differentials between different packets on the same flow.

בחלק "Feature Extraction" - II" - ניתן לראות שהמאמר מתאר כיצד החילוץ מאפיינים מכל Session של תעבורת רשת (HTTPS). ראשית נראה שתעבורה מוצפנת מסתכמת על פרוטוקול SSL/TLS עבור תקשורת מוצפנת, פרוטוקולים אלה בנויים מעל השכבה של הפרוטוקולים TCP/IP.

השכבה בTCP- תקבל את הנתונים המוצפנים מהשכבה העליונה (שבה SSL/TLS נמצאים). ותחלק אותם לקטעים במידה וגודל החבילה עובר את ה-(MSS (Maximum Segment Size. עבור כל קטע כזה,

פרוטוקול TCP יוסיף Header ובכך ייווצר סגמנט.

כל סגמנט כזה נכנס לתוך הDatagram של פרוטוקול IP. בנוסף, מאחר וחבילות של TCP לא יכללו מזהה ייחודי עבור, ולכן מחברי המאמר בחרו לזהות את הסשן בעזרת הtuple- אשר יכלול את - הפרוטוקול עצמו, כתובת המקור, כתובת יעד, פורט המקור, פורט היעד.

החלוקה של כל סשן מתבצעת באופן הבא - הסשן יחולק לשני זרמים - זרם קדמי (Forward), זרם אחורי (Backward). הזרם מוגדר כך שהוא מבטא רצף מסודר בזמן של חבילות TCP עבור סשן יחיד של TCP.

הזרימה הקדמית מייצגת סדרה של חבילות שמגיעות בלבד, בעוד שהזרימה האחורית מייצגת חבילות אשר

יוצאות. שיטה זו תאפשר הצגת של נתונים בין שני הצדדים, ומשמש כהצגה של חיבור.

בנוסף, מחברי המאמר עושים שימוש במאפייני סדרות של זמנים כגון הפרש זמני ההגעה של חבילות מאותו זרם וכו'..

בטבלה, קל לראות את כל המאפיינים הללו באופן ברור -

Forward packets אשר מייצג את מספר החבילות הנשלחות בזרם הקדמי, כפי שצוין מקודם.

Backward packets - מייצג את מספר החבילות הנשלחות בזרם האחורי, כפי שצוין מקודם.

Forward/backward total bytes - כלומר סך הבתים אשר נשלחו בזרם הקדמי/אחורי.

min forward/backward inter arrival time diff - שצוין כבר מקודם, עבור שני הזרמים.

max forward/backward inter arrival time diff - הפרש הזמן המקסימלי בין שתי חבילות בזרימה הקדמית/אחורית.

mean forward/backward inter arrival time diff - ממוצע הפרשי הזמנים בין הגעת חבילות עוקבות בזרימה הקדמית/ אחורית.

STD forward/backward inter arrival time diff - סטיית תקן של הפרשי הזמנים בין חבילות עוקבות בזרם הקדמי/אחורי.

mean forward/backward packets - הממוצע של מספר החבילות בכל קבוצה/חלוקה פנימית בזרם הקדמי/ אחורי.

STD forward/backward packets - סטיית התקן של מספר החבילות בכל קבוצה/ חלוקה פנימית בזרם הקדמי/ אחורי.

mean forward TLL value - ערך ה-Time to live הממוצע של החבילות בזרם הקדמי.

min/max forward/backward packet - הגודל המינימלי/מקסימלי של חבילה בזרימה הקדמית/ אחורית.

total packets - מספר החבילות הכולל (של זרם קדמי ואחורי) בסשן כולו.

min/max packet size - הגודל המינימלי/ מקסימלי של חבילה בסשן כולו ללא תלול בכיוון הזרם.

mean packet size - ממוצע גודל החבילה בסשן באופן כולל.

packet size variance - השונות של גדלי החבילות בסשן באופן כולל.

ניתן לראות שמאפייני בסיס אלה מספקים מדדים סטטיסטיים אשר מכסים הרבה מדדים נחוצים. זאת במטרה לאפשר סיווג מדויק של תעבורה מוצפנת.

כעת נעבור למאפיינים החדשניים -

בנוסף למאפייני הבסיס, מחברי המאמר מציעים סט חדשני של מאפיינים שמטרתם לשפר את יכולת הסיווג ובכך להבחין בצורה מדויקת את סוגי התעבורה השונים. סט זה מתבסס על מאפייני SSL, מאפייני TCP ובנוסף את ההתנהגות המתפרצת של האתרים אשר מתבטא בחלקים של התעבורה כאשר יש "שקט" לפני ואחרי, חלק זה נקרא "השיא" (Peak).

נפרט על מאפיינים חדשנים אלה -

ישנו שימוש במאפייני TCP מורכבים כגון -

TCP initial window size ובנוסף TCP window scaling factor , אשר יבטא את אופן ניהול חלון השליחה של החבילות והקצאת משאבים בפרוטוקול TCP.

בנוסף לכך, יש שימוש במאפייני SSL/TLS מתקדמים אשר יספקו מידע על שיטות הדחיסה, הרחבה, הצפנה של כל סשן בפרוטוקול אשר מוזכרים בטבלה.

SSL compression methods, SSL extension count, SSL cipher methods, SSL session id .length, Forward SSL Version

הזכרנו למעלה את השימוש במאפיין ההתנהגות המתפרצת של הדפדפן, מאפיינים אלו יתארו מצבי "שיא"

בתעבורה, כלומר יכול להיות מצב שבו קטעים מסוימים

יהוו פעילות גבוהה בתעבורה ולאחר מכן תיהיה שתיקה

שתתבטא באופן יחסי. להלן צירוף של צילום מסך מהמאמר הממחיש זאת. במאמר מוזכר שימוש ייחודי במאפיין זה לדפדפן כאשר נדרש טעינת דפים מסוימים או ייצוג של תכני מולטימדיה.

לפי הטבלה, המחברים סיפקו מדדים אשר ישקפו ההתנהגות המתפרצת במגוון אופנים כגון -

ספירה של מספר הקטעים בזרימה אשר בהם מתרחש

"Peak", כלומר תעבורה אינטנסיבית בזרם הקדמי/אחורי.

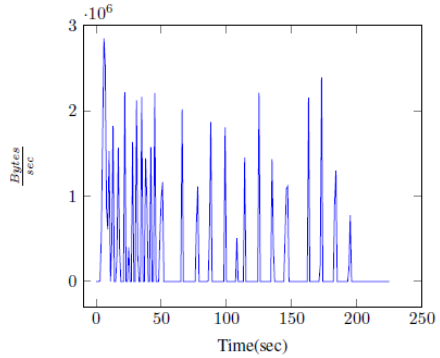


Fig. 3: An example of the bursty behavior of browser traffic.

ולבסוף ישנם גם מדדים שנעשו באופן דומה במאפייני הבסיס. כלומר ישנה הצגת הסטטיסטיקה של קצב ההעברה בזמני התעבורה האינטנסיבית.

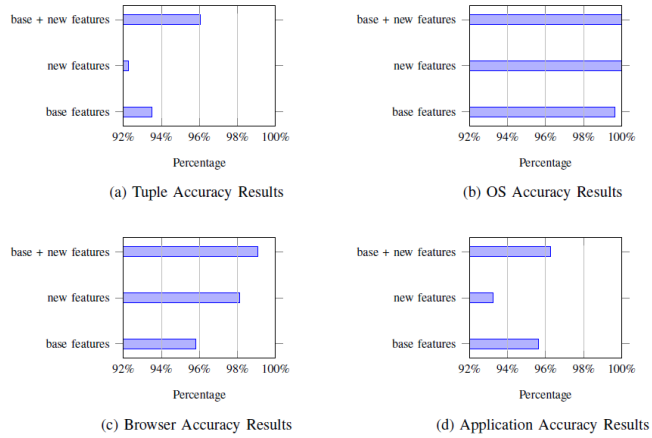
ובנוסף חישוב של הפרשי זמני הגעה של החבילות במהלך כל זמן תעבורה אינטנסיבי.

לקראת הסוף ניתן לראות שימוש במאפיינים נוספים כגון -

חבילות "Keep Alive" - מידע על התנהגות הפרוטוקול ואופן הניהול של החיבורים.

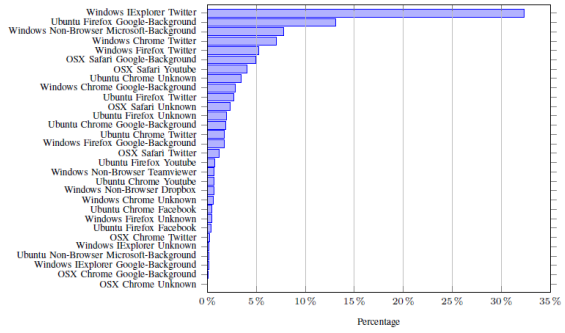
TCP Maximum Segment Size - גודל החבילה המקסימלי ברשת.

להלן תוצאות המחקר ומסקנותיהן -Screenshots are in the following order - Figure 2, Figure 1 Figure 4. These showcase the Article's Results.

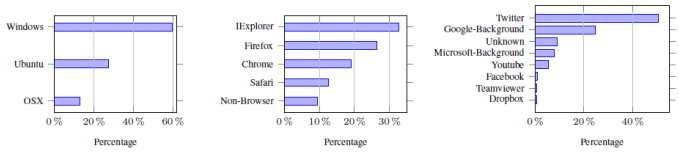


(a) Tuple Accuracy Results (b) OS Accuracy Results (c) Browser Accuracy Results (d) Application Accuracy Results

<u>Figure 1</u> - תוצאות הסטטיסטיקה של Dataset -t. באמצעות השיטות למידת מכונה שבה המחברים השתמשו, הם הגיעו לתוצאות משמעותיות. ניתן לראות מהתצלום תוצאות סיווג עבור סט מאפיינים שונים. קל לראות שהמחברים אכן הצליחו להראות שאכן הרשומת סיווג <OS,Browser,Application> אכן אפשרית עם סיכוי להצלחה גבוה. נשים לב כי שיטת הסיווג הנאיבית קיבלה אחוז דיוק של 32.34% לבסוף ניתן לראות שבנוסף למאפיינים התעבורה הבסיסיים, ההוספה של המאפיינים החדשים הצליחה לעלות את אחוז הדיוק ל96.06%.



(a) Dataset labels (tuple) statistics



(b) Dataset OS statistics (c) Dataset browser statistics (d) Dataset application statistics

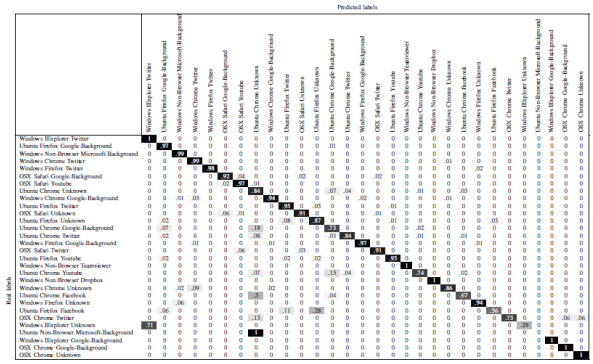
Fig. 1: Dataset statistics

<u>Figure 4</u> - מחברי המאמר סיפקו תוצאות באמצעות מטריצות בלבול. ניכר לראות שתוצאות הבדיקות הן אכן כמעט מושלמות. סיווג מערכת ההפעלה נעשה ללא שגיאות, סיווג הדפדפנים נעשה ברמת דיוק גבוהה מאוד עם מעט שגיאות, דוגמה לכך היא שגיאה בזיהוי אפליקציה כמו Facebook, שלעיתים הייתה מסווגת בתור "Unknown", שהתחרש בכ-29% מהמקרים, בעוד שרוב האפליקציות סווגו בדיוק גבוה ובטוח. מקרה נוסף של שגיאה התרחש כאשר tuple סווג

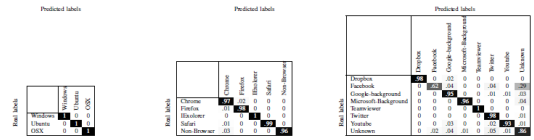
בתור "Ubuntu Chrome Unknown" מה שקרה ב-18% מהמקרים או בתור - " Ubuntu Firefox Google-Background" ב-7% מהמקרים, והוא הוגדר כ - "Ubuntu Non-Browser Microsoft-Backround". באופן כללי, נתונים אלו נותנים אור על יכולת סיווג יוצאת דופן עם שיפור ניכר. לעיתים נוצרים שגיאות אשר נוטים להתרחש במקרים בהם הקטוגריה היא אינה ודאית או אינה חופפת.

מסקנות -

במסגרת המחקר ושיטת העבודה, ניתן אכן להגיד כי המאמר מספק הוכחה לכך שאכן ניתן לסווג תעבורה מוצפנת ברשת, ובכך לזהות ולהסיק באיזה מערכת הפעלה, דפדפן ואפליקציה נמצא המשתמש במחשב הנייח/נייד שלו מה שמדגיש את הפער שבין התעבורה המוצפנת לבין שמירת הפרטיות המלאה של הגולש. המחקר בהחלט מדגים כי למרות השימוש בהצפנה של TLS/SSL, גישת הניתוח התעבורה שמוצגת בפנינו יכולה לאפשר לתוקף פוטנציאלי לנצל זאת לטובתם. בין אם זה לזיהוי משתמש כלשהו או לצורך אסטרטגי. בנוסף המחקר מדגיש כי ניתן להרחיב את המחקר בכך שניתן להוסיף פעולות ספציפיות כגון שליחת ציוץ או קבלת פוסט לצורך סיווג, ואף מוצע להעביר את המחקר לסביבה של מכשירים ניידים וככה להעמיק את הפוטנציאל שטמון ברעיון המחקר.



(a) Tuple Confusion Matrix



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

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

רקע :

סיווג תעבורת אינטרנט הוא תחום מחקר פופולרי בגלל הצורך בשיפור איכות השירות, הנדסת רשת, אכיפת חוק ואפילו בזיהוי תוכנות זדוניות. עם העלייה בשימוש בהצפנות ובכלים כמו VPN ו-Tor, הסיווג הפכה למשימה קשה יותר . שיטות תעבורה מוצפנת הסתמכו לרוב על ניתוח התוכן של חבילות או על תכונות ידניות סטטיסטיות שהוזנו למודלים של למידת מכונה שטחיים כמו SVM, Decision Trees ו-KNN. הגישות אלו התקשו להתמודד עם הצפנות מודרניות ופגעו בפרטיות של משתמשים והציגו נתונים לא מדוייקים.

המאמר מציע פתרון לבעיה זו על ידי הפיכת זרימת רשת לתמונה בשם FlowPic ושימוש ברשתות נוירונים קונבולוציוניות המוכרות בתחום זיהוי תמונות. טכניקה זו מבוססת על תכונות מבוססות זרימה בלבד, כגון גודל וזמן הגעת החבילות, ומאפשרת סיווג מדויק של סוגי תעבורה (גלישה, צאט) ושל יישומים ספציפיים גם בסביבת הצפנה מורכבת. המאמר מתמקד בשימוש במאגרי נתונים של ISCX ובוחן את ביצועי השיטה במצבי הצפנה שונים, תוך השוואה לשיטות קודמות והדגשת היתרונות שבשימוש ב-CNN במודל גנרי וגמיש.

<u>התרומה העיקרית של המאמר :</u>

המאמר מביא תרומה חדשנית בתחום סיווג תעבורת האינטרנט המוצפנת. המאמר מציג שיטה ייחודית שבה נתוני זרימת הרשת מומרת לתמונה FlowPic ולאחר מכן משתמשים בטכניקות מתקדמות של למידת מכונה באמצעות רשתות נוירונים קונבולוציוניות כדי לסווג את סוג התעבורה ואת היישום הפעיל.

FlowPic מציע גישה המתבססת על נתונים מבוססי זרימה בלבד Packet Size ו- Normalized Arrival Time תהליך זה שומר על פרטיות המשתמשים בגלל שהוא אינו דורש גישה למידע הפנימי של החבילות אלא רק לתכונות סטטיסטיות כמו גודל החבילה וזמן ההגעה שלה.

היתרון המרכזי בא לידי ביטוי ביכולת המודל להתמודד עם סוגים שונים של הצפנה, כולל VPN ו Tor שמקשים על סיווג תעבורה בשיטות אחרות. המאמר מציג תוצאות עם דיוק של עד 99.7% בסיווג יישומים ספציפיים, ודיוק של מעל 98% בסיווג תעבורה מוצפנת ב-VPN. אף על פי שבמצבי הצפנת Tor הדיוק היה נמוך יותר כ-67.8% כאשר המודל אומן במפורש על תעבורת Tor, הדיוק עלה באופן משמעותי דבר המדגים את גמישותו של המודל ואת יכולתו ללמוד דפוסים מורכבים.

מעבר לביצועים הטכניים השיטה מציגה גישה גנרית במגוון תחומים כמו אבטחת רשתות, זיהוי תוכנות זדוניות, שיפור ניהול רשתות ואכיפת חוק. השימוש במודל CNN אחיד לכל סוגי הבעיות, ללא צורך בהתאמות פרטניות, מאפשר ליישם את השיטה בקלות במערכות קיימות ולשפר את היכולת שלהן להתמודד עם תעבורה מוצפנת המשתנה באופן תדיר.

לסיכום, התרומה המרכזית של המאמר היא בהצגת פתרון חדשני, מדויק וגמיש לסיווג תעבורת אינטרנט מוצפנת, המשלב בין עקרונות של עיבוד תמונה למערכות רשתות

תקשורת, ותורם להתקדמות משמעותית בתחום האבטחה והניתוח של תעבורת רשתות בעידן ההצפנה המודרנית.

מאפייני התעבורה של המאמר:

במקום להשתמש בתכונות ידניות או בניתוח תוכן החבילות, המאמר מתמקד במאפיינים שמבוססים רק על זרימה , נתוני זמן ההגעה של החבילות וגודל החבילות . החידוש שמציג המאמר הוא בניית התמונות FlowPic המייצגת את דיאגרמת הנתונים הדו ממדית, המשלבת גם את גודל החבילות וגם את זמני ההגעה, ויצירת מודל למידה אחיד שמתאים לכל בעיות הסיווג שהוצגו.

התוצאות המרכזיות של המאמר והתובנות:

המאמר מציג תוצאות טוב בשימוש FlowPic לסיווג תעבורת אינטרנט מוצפנת. המודל השיג דיוק של 85% בסיווג קטגוריות תעבורה לא מוצפנת, ודיוק גבוה במיוחד של 98.4% בתעבורה מוצפנת ב-VPN. גם בתנאי הצפנה מורכבים יותר, כמו תעבורה דרך Tor, הצליח המודל להגיע לדיוק של 67.8%. בסיווג יישומים ספציפיים השיטה הציגה ביצועים מצויינים עם דיוק של 99.7%. התובנות המרכזיות מהתוצאות מדגישות את יכולת המודל לשמור על דיוק גבוה גם במצבי הצפנה, את היכולת לזהות יישומים חדשים שלא היו בשלב האימון, ואת יתרונו בשמירה על פרטיות המשתמשים ללא צורך בניתוח תוכן החבילות. המאמר מדגים כי FlowPic מביאה לשיפור משמעותי לעומת מחקרים קודמים, בכך שהיא משתמשת במודל רשת נוירונים קונבולוציונית אחיד, ללא צורך בהתאמות פרטניות, ובכך פותחת אפשרויות חדשות בתחום סיווג תעבורה מוצפנת ברשתות תקשורת.

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

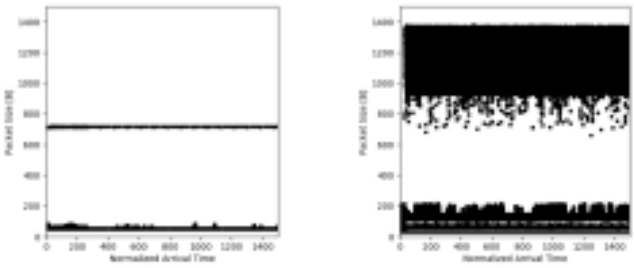
במאמר מוצגת הטבלה הזאת שמציגה את אחוזי הדיוק של מודל FlowPic בסיווג קטגוריות שונות של תעבורת רשת תחת שלושה מצבי הצפנהNon-VPN, VPN ו Tor הקטגוריות כוללות

סוגי תעבורה מגוונים, כל שורה מציגה את אחוזי הדיוק כאשר המודל אומן ונבחן בתנאי הצפנה שונים.

כפי שהמאמר מציין המודל מצליח לשמור על דיוק גבוה גם במצבי הצפנה נפוצים כמו VPN לעומת זאת, במצבי הצפנת Tor המודל מציג ירידה משמעותית בדיוק למשל בקטגוריית VoIP הדיוק עומד על 48.2% בלבד כאשר המודל אומן על Non-VPN, אך משתפר ל-93.3% כאשר המודל אומן . המאמר מציין כי הצפנת Tor יוצרת דפוסי חבילות אחידים המקשים על זיהוי דפוסים ייחודיים, והטבלה מראה את הטענה הזו באמצעות התוצאות המעידות על הצורך באימון ייעודי למצב זה.

התוצאות המוצגות בטבלה משקפות את המסר המרכזי של המאמר FlowPic הוא מודל גמיש ויעיל שיכול להסתגל לתנאי הצפנה שונים, במיוחד כאשר מתבצע אימון מתאים לכל סוג הצפנה. בנוסף, הטבלה מדגישה את החוזק של המודל בסיווג סוגי תעבורה מסוימים כמו Video ו Chat גם במצבי הצפנה מורכבים .

פה ניתן ליראות איך עובדת השיטה בהשוואה הראשונה רואים 2 תמונות FlowPic של Netflix ו- Skype ורואים ברור איך אפשר להבדיל בניהם בעזרת בתמונה . תמונה A מראה תבנית אחידה של גודל חבילות קבוע יחסית שזה מאפיין סטרימינג . ובתמונה B רואים את הפיזור של גודל החבילות שזה דבר שהגיוני ל SKYPE כי יש שם גם ודאו וגם אודיו ויכול להיות גם הודעות צאט , כך שלכול חבילה יש גודל שונה.



(a) Netflix (b) Skype

Part III - Section 1

בחלק זה נציג את שיטת העבודה שלנו כלפי ההקלטות של התעבורה -
בחרנו להקליט כל אפליקציה באורך כ- 3 דקות

1. גלישה בדפדפן Google-Chrome באתר אינטרנט CNN - גלשנו באתר החדשות הידוע - www.CNN.com, במהלך הגלישה ביצענו מעבר בין מאמרים שונים.
צפי ההקלטות - אנחנו כמובן נצפה להרבה פקטות TCP, כאשר רוב התעבורה תיהיה מוצפנת באמצעות HTTPS (כלומר - ניתקל בפקטות TLS רבים). בשל המעבר שלנו במאמרים שונים באתר החדשות, תתרחש תעבורה רבה ולכן באמצעות ניתוח הקלטה זו באמצעות הקוד האוטומטי שלנו, נוכל בקלות לראות כיצד התעבורה באתר זה אכן מתרחשת.

3. עבור Audio-Streaming בחרנו כמובן באפליקציית המוזיקה החביבה עבור כלום - Spotify.
נכנסו לאתר הראשי של ספוטייפי - www.spotify.com, דרך גוגל כרום ולא דרך האפליקציה על מנת שלא תיהיה בעיית חסימה למפתחות TLS כי הרי אנחנו רוצים לנתח את התעבורה בצורה מדויקת. והדלקנו שמיעה של שיר מסוים בתוך אתר זה.

4. עבור Video-Streaming בחרנו להשתמש ב-www.youtube.com. לאפליקציה זו נכנסו דרך אתר הדפדפן Google-Chrome ובו צפינו בסרטון וידיאו באיכות 1080P.
וידיאו באיכות שכזו יצרוך עבורנו רוחב פה משמעותי ולכן נצפה לראות פקטות גדולות בנוסף לתדירות שליחה גבוהה.

5. עבור Video-Conferencing בחרנו להשתמש ב-Google Meet
באפליקציה זו ביצענו שיחה קבוצתית עם וידיאו וקול למשך כ-3 דקות על מנת לצבור תעבורה תקנית ורלוונטית לניתוח.

בחרנו לנתח את התעבורה באפליקציות השונות באמצעות מספר מאפיינים, נציג אותם ולאחר מכן נספק מידע כיצד הם יעזרו לנו בניתוח.

חשוב לציין שבכל הקלטה שלנו אנחנו ניסינו כמו שיותר לצמצם רעשים בכך שפעלנו לפי ההוראות שדרשו מאיתנו במטלה - (לכבות התראות וכו'), בנוסף לכך, בקוד פייתון שלנו, כאשר אנו מבצעים אוטומציה לניתוח הקלטת תעבורה מסוימת, אז אנחנו מבצעים סינון יעיל אשר לא רק יעזור לנו לפענח את הפקטות החיוניות לנו אלא גם משפר יעילות בזמני הריצה של ניתוח ההקלטה.

1. IP Header Fields
- TTL (Time To Live)
למדנו בקורס שTTL הינו מספר הקפיצות המקסימלי שחבילה יכולה לעבור לפני שהיא "נמחקת" מהרשת.
בחרנו להציג את הערך הזה על מנת להבין אילו

2. TCP Header Fields - עבור פרוטוקול זה בחרנו להתמקד בשני מאפיינים עיקריים -
Window Size - בעזרת מאפיין זה נוכל לדעת עד כמה בתים של מידע אנחנו יכולים להעביר לפני שנקבל ACK. מה שמשפיע כמובן על מהירות התעבורה.

TCP Flag (דגלי ה- TCP)- באמצעות הדגלים נוכל לבחון על אופן ניהול החיבורים בין האפליקציות השונות.

3. TLS Header Fields - בחרנו להתמקד בשלושה מאפיינים לניתוח התעבורה המוצפנת HTTPS.
TLS Record Length - נוכל לדעת על אורך הרשומה של הפקטה המוצפנת ובכך להבין יותר את מבנה התעבורה באפליקציה.
TLS Content Type - נוכל לזהות את סוג המידע שמועבר לנו בתוך הפקטות TLS, חילקנו את זה לשני סוגים אפשריים - Application Data/Handshake, אם למשל יהיו לנו כמויות גדולות של לחיצות ידיים אז נוכל להסיק שמדובר בחיבורים קצרים כגון גלישה באתר אינטרנט. לעומת זאת אם נראה הרבה סוגים של Application Data אז נוכל להסיק שיש העברה של נתונים בצורה רציפה, מה שיכול לרמוז על אפליקציות סטרימינג ושיחות וידיאו.
TLS Version - מאפיין זה מאפשר לנו לבדוק באיזה גרסת TLS הפקטה נמצאת. עבור מאפיין זה ניתן לתת אור עבור רמת האבטחה של אותה אפליקציה, אנו יודעים שכיום רוב האפליקציות עם גרסאות TLS 1.2/TLS 1.3 אך לצורך הרחבה ואבחון תעבורה רצינו לצרף מאפיין זה גם.

4. Packet size - מאפיין זה מאפשר לנו לבחון את גודל כל פקטה בהקלטות שלנו. בעזרת מאפיין זה נוכל לזהות הבדלים עיקריים בין כל אפליקציה.

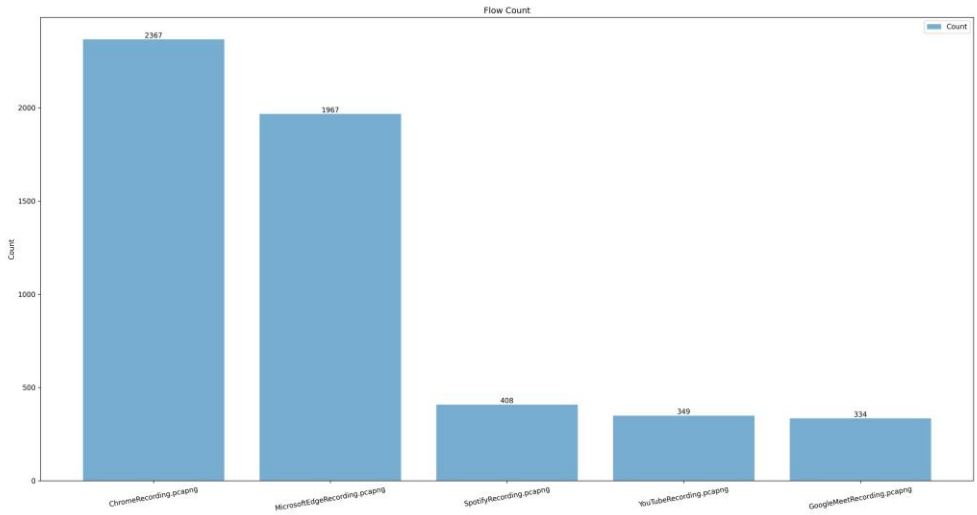
5. Flow Size - מאפיין זה מייצג את סדרת הפקטות בין המקור ליעד לאורך חיבור ספציפי.
זרמים קטנים יעידו לנו על חיבורים קצרים, נצפה למקרים כאלה בעת גלישה באתר מסוים בדפדפן כאשר יש טעינה של דף,תמונות, פרסומות וכו'.
זרמים גדולים יעידו על אפליקציה שדורשת חיבור רציף וחזק, מה שירמוז לנו על אפליקציית סטרימינג כמו YouTube או אפליקציית שיחות כמו Zoom,Google-Meet.

6. Time difference between packets received - בעזרת מאפיין זה נוכל לבחון באיזה תדירות אפליקציה מסוימת תשלח פקטות רלוונטיות. למשל נוכל לראות האם יש קצב שהוא עקבי בעת צפייה בסרטון ביוטיוב או אם יש פיזור שהוא אחיד בעת מעבר בין מאמר למאמר באתר אינטרנט כלשהו.

7. Protocol Counts - באמצעות מאפיין זה נוכל לראות כמה את כמות הפקטות בכל הקלטה ששייכות לפרוטוקולים הרלוונטים, כך נוכל לזהות יותר טוב ומדויק את סוג האפליקציה בתעבורה.

Section 2-3 - Plot Comparisons

הבא נתחיל בהשוואת בין ההקלטות.



1. <u>ניתוח גרף לפי מאפיין Flow Count -</u>

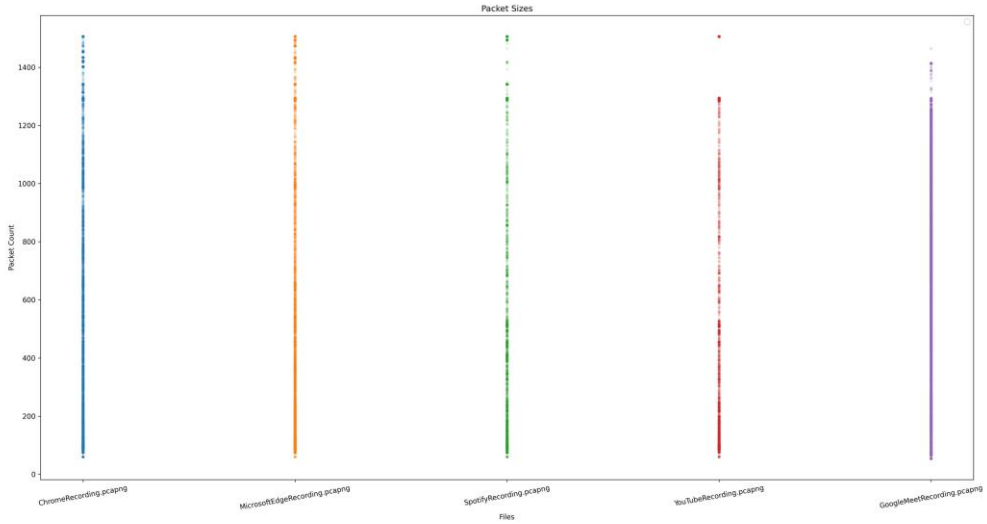
בגרף זה ניתן לראות את המספר הזרמים שנוצרו במהלך כל הקלטה. נתון זה חשוב להבנת אופן פעולת התעבורה מאחר והוא מייצג כל קבוצה של פקטות שמתקשרות ביניהן. מספר זרמים גבוה יכול להעיד עבורנו שהאפליקציה שלנו יצרה מספר רב של חיבורים קטנים - נוכל לראות זאת עבור הקלטות הדפדפנים Google-Chrome ו-Microsoft-Edge, כאשר בהם גלשנו באתר החדשות CNN ובו עברנו בין מאמרים שונים באתר, כאשר כל מעבר כזה עשוי לפתוח חיבור חדש. לעומת זאת, נבחין שאפליקציות סטרימינג של וידאו או שירים עשויה להציג מספר זרמים מצומצם יותר אך עם נפח גבוה יותר לכל זרם.

נראה לפי הגרף שאכן יש הבדל מסוים בין הדפדפנים Google-Chrome ו--MS edge, בדפדפן Chrome נוצרו כ- 2367 זרמים, בעוד שב- MS-Edge נוצרו כ- 1967 זרמים.
כאשר הסבר הגיוני עבור זה יכול להיות מאחר שבדפדפן מסוים עברנו ביותר מאמרים ובכך ביקשנו יותר משאבים, או שדפדפן אחר טען יותר תוכן רקע, הרחבות וכו'.

עבור אפליקציות Audio Streaming, Video Streaming, Video conferencing (Youtube,Spotify,Google-Meet), ניתן לראות לפי הגרף שכמות הזרימות שלהם במהלך ההקלטות הוא נמוך משמעותית לעומת שני הדפדפנים שלנו כאשר -
ניתוח הקלטת אפליקציית Google - Meet הביאה כ- 334 חיבורי זרמים.
ניתוח הקלטת אפליקציית YouTube הביאה כ- 349 חיבורי זרמים.
ניתוח הקלטת אפליקציית Spotify הביאה כ- 408 חיבורי זרמים.

קל להבחין כי כמות הזרמים באפליקציות אלה היא נמוכה משמועתית מכמות הזרמים בדפדנים. מכך נסיק כי אפליקציות סטרימינג אלה יוצרות חיבורים ש"חיים" לאורך זמן, כלומר נשארים פתוחים לכל אורך סשן מסויים, כאשר זה מתבטא לצורך הזרמה רציפה של קול שמע או וידיאו. בזום למשל כאשר נרצה לפתוח שיחת פגישה, נדרש חיבור יציב להעברת הקול בזמן אמת ולבקרה, וללא פתיחה של חיבורים חדשים בכל מעבר בין עמור או משאבים חיצוניים.
לסיכום - ראינו שהדפדפנים Chrome ו-Edge יוצרים מספר רב של זרמים קצרים, מה שמצביע על טעינת משאבים מרובים בכל מעבר בין מאמר למאמר (פרסומות, תמונות וכו') בעוד שאפליקציות סטרימים כמו (YouTube, Spotify) ואפליקציה כמו-Google Meet יוצרות זרמים מצומצמים יותר בכך של זרם מכיל בתוכו נפח נתונים שהוא גדול יותר והוא גם חי לאורך זמן, דבר זה מתבטא ביציבות והאופן הרציף של ההזרמה.

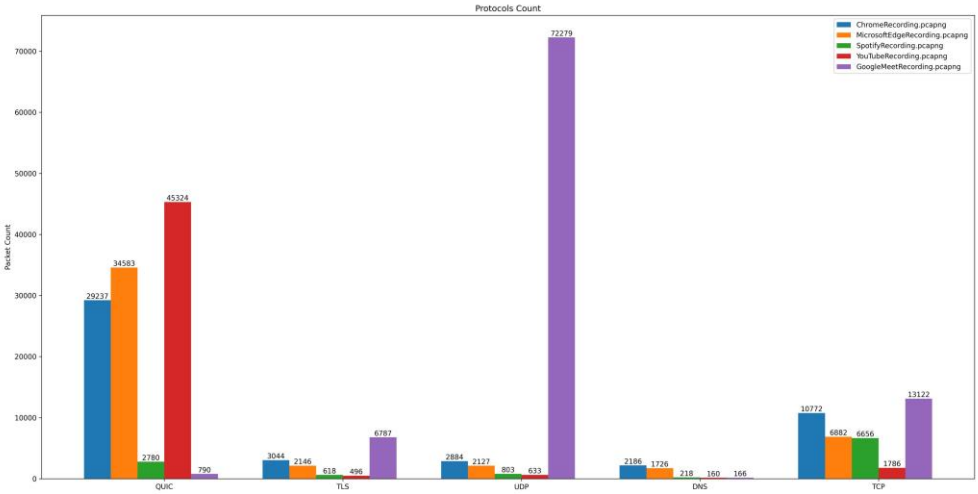
2.ניתוח גרף לפי מאפיין-Packet Size



הגרף מציג גודל מנות (Packet Sizes) עבור כל קובץ הקלטה. כל צבע/עמודה מייצגת קובץ הקלטה אחר, והנקודות לכל אורכה מראות את ההתפלגות של גדלי המנות באותו קובץ. כל נקודה על הגרף מייצגת גודל של פאקטה ספציפית בהקלטה מסויימת.

מהגרף אפשר לראות שגלישה חופשית בדפדפנים (Chrome, Edge) ושיחות ווידאו נותנות מגוון של גדלי פאקטות. לשירותי סטרימינג (YouTube, Spotify) יש גדלי פאקטות שדי מקובצות לגדלים נמוכים ובאותו סדר גודל.

3. ניתוח גרף לפי מספר כמות הפרוטוקולים עבור כל הקלטה -



גרף זה מייצג את כמות הפרוטוקלים שנעשו בהן שימוש במהלך כל הקלטה, החלטנו לייצג חמישה פרוטוקולים עיקריים והם - QUIC, UDP, TLS , TCP, DNS. המטרה העיקרית עבורנו היא לסנן תעבורה שהיא אינה רלוונטית אלינו.

<u>עבור פרוטוקול QUIC</u> - ניתן לראות שמתקיימת תעבורה רבה בהקלטות עבור הדפדפנים MicrosoftEdge, Google-Chrome וכן בהקלטה עבור YouTube. אנו יודעים כי QUIC מבוסס על פרוטוקול UDP. ולכן כאשר גלשנו בדפדפנים באתר CNN, הייתה עדיפות עבור הדפדפנים להשתמש בQUIC על מנת לשפר את הטעינה של דפי האינטרנט, וכן עבור YouTube כדי לשפר את טעינת הווידיאו באיכות הגבוהה שבחרנו לצפות.

<u>עבור פרוטוקול UDP</u> - ניתן לראות שיש שימוש רב בפרוטוקול UDP עבור Google-Meet כמו איך שציפינו אליו, מאחר ואפליקציה זו מתבסס על תקשורת בזמן אמת, אין לו חשיבות רבה עבור אובדן פקטות אלא הוא נדרש לביצועים ולכן הגיוני מאוד שנראה כמות גדולה של פקטים מפרוטוקול זה עבור אפליקציה זו. לעומת זאת, עבור אפליקציות YouTube ו- Spotify נראה כי יש יותר שימוש בפרוטוקול QUIC לצורך

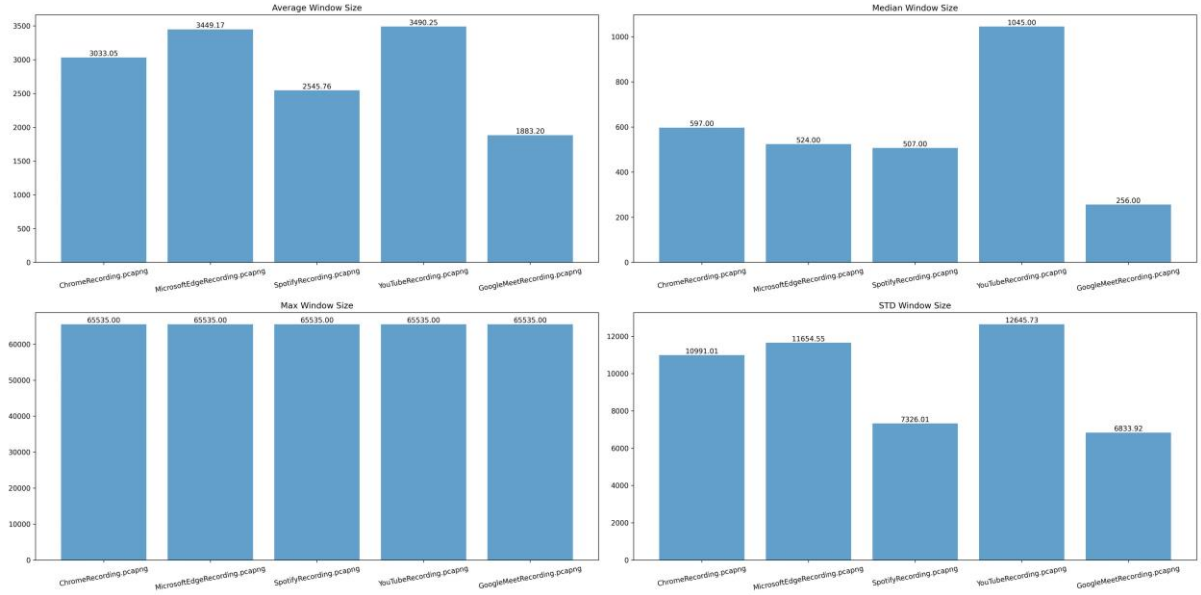
שיפור קצב ההעברה.

<u>עבור פרוטוקול TLS</u> - אשר מייצג תקשורת מאובטחת , כלומר HTTPS מעל TCP.
ניתן לראות אכן שיש שימוש בפרוטוקול זה, יש לציין שאף עבור כל הקלטה שמרנו מפתחות TLS לצורן ניתוח מאפיינים עבור הHEADER שלו.
ניתן לראות ובצדק שתהיה כמות יותר גבוהה של פקטים מפרוטוקול זה עבור הדפדפנים מאחר ואנחנו גולשים באתר חדשות CNN ועוברים בין מאמר למאמר, פעולות אלה טוענים משאבים רבים ולכן בנוסף זקוקים להצפנה בHTTPS.
לעומת זאת, עבור Spotify נראה שכן קיימת תעבורה אך היא נמוכה יחסית להקלטות הדפדפנים, אכן אנחנו נכנסים לדפדפן Google לאתר Spotify, ולכן תיהיה הצפנה של HTTPS.
עבור YouTube, נראה כי גם פה יש כמות קטנה של פרוטוקולים אלה, מאחר ונעשה בו יותר שימוש בQUIC, ניתן להגיד שגם בהקלטה זו מאחר והייתה כניסה לאתר דרך הדפדפן שכן תדרוש הצפנה ובנוסף זה יכול להעיד על משאבים אחרים (פרסומות וכו').
Google-Meet, כפי שציינו מקודם ראינו הרבה פקטות TCP, פה בנוסף היינו צריכים להתחבר לאתר ולכן ידרשו פה פקטות TLS.

<u>עבור פרוטוקול TCP</u> - ניתן לראות כי עבור הקלטות הדפדפנים יש מספר פקטות רב מסוג פרוטוקול זה, אולם ניתן לראות הבדל קטן בין כמות הפרוטוקולים, דבר המצביע על אופן ההקלטה, כנראה באחד מהם ניגשנו ליותר מאמרים בפרק זמן ההקלטה, בנוסף זה יכול להצביע על אופן הפעולה של אותו דפדפן והטעינה של התכנים. בנוסף , נראה שכאשר נכנסו לSpotify דרך דפדפן Chrome, ניתן לראות כי גם בהקלטה זו יש נוכחות רבה של פרוטוקולים אלה, מה שמחזק את הטענה שיכול להיות שדפדפן גוגל מעדיף להשתמש ביותר פקטות TCP.

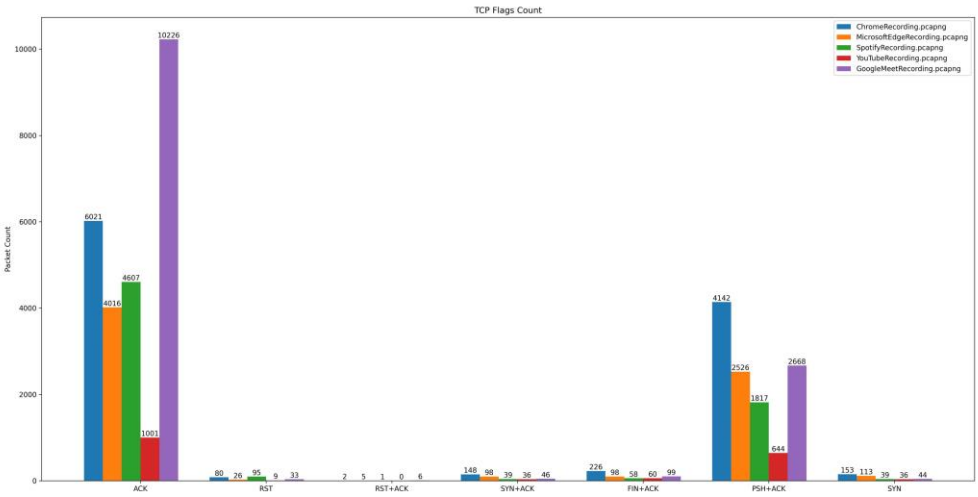
<u>עבור פרוטוקול DNS</u> -
נראה שבכל האפליקציות יש שימוש בDNS, כאשר הדפדפנים Google-Chrome, MicrosoftEdge מציגים שאילתות של DNS בכל מעבר של דף חדש, שבו יכולים להיטען כתובות, פרסומות, תמונות וכו'.

4. ניתוחים סטטיסטים עבור מאפיין TCP Header-



גודל חלון קובע עבורנו כמה בתים צד מסוים יכול לשלוח לפני שנדרש אישור -ACK.
אנו יודעים כי חלון גדול מסמל עבורנו שיש ניצול מעולה של הרשת ויש קצב העברה מעולה,
אך עם יותר סיכונים לאיבוד מידע.
ולעומת זאת, אם גודל החלון קטן אזי יש פחות סיכון לאבד מידע, עבור אפליקציות רגישות נראה זאת לפי הגרפים. ניתן לראות שעבור אפליקציית YouTube יש חלון בגודל ממוצע של 3490.25, כלומר ניתן להסיק שבעת ההקלטה התרחשה זרימה רציפה של פקטים לטובת יציבות הצפייה בסרטון. לעומת זאת, נביט בגודל החלון הממוצע של אפליקציית Google-Meet ניתן לראות שהוא יחסית קטן לעומת YouTube,עם חלון ממוצע בגודל 1883.20. כלומר ניתן לראות תעדוף שח רוחב פס גבוה על מנת שתיהיה התאמה טובה עבור שיחות וידיאו. נוכל לראות שאפליקציית Spotify מציג גודל חלון ממוצע של 2545.76 כלומר יש תעבורה רציפה, אך בנפח נמוך יותר מאשר אפליקציית YouTube.
בנוסף, הקלטות הדפדפנים Google-Chrome ו-Microsoft-Edge מציגים ממוצעים כמעט קרובים. Microsoft-Edge עם ממוצע 3449.17 לעומת 3033.05 של Google-Chrome.

מאפיין נוסף עבור TCP Header יהיה אוסף הדגלים עבור כל הקלטה -



מאפיין נוסף שאפשר לקחת לגבי TCP Header זה כמות הדגלים:

נסקור את הגרף הדגלים:

ACK : הערכים הגבוהים ביותר שייכים ל-Google Meet (10,226 חבילות) ולאחר מכן Chrome (6,021). ACK מופיע בכל סגמנט שמתקבל בהצלחה, כך שכמות גבוהה מצביעה על תעבורה מרובה או סגמנטים קטנים ורבים. כלומר ניתן להסיק שתעבורת גוגל מיט שולחת חבילות קטנות יותר. לעומת זאת יוטיוב עם מעט מאוד ACK, מה שניתן להסקה הוא שכנראה האפליקציה מעדיפה לשלוח מידע בגושים גדולים של מידע.

RST (איפוס חיבור): נשים לב שהבולטים עם מספר יחסית גבוה של RST הם כרום וספוטיפיי. מה שמעיד על חיבורים זמניים רבים (אולי לפרסומות או למטא-דאטה) שנחתכים ב-RST.

RST+ACK מצביע על סגירה פתאומית, אך תוך מתן ACK אחרון. הכמות הנמוכה יחסית של "RST+ACK" מצביעה שהמקרים האלה אינם התרחיש השגרתי – ברוב המקרים, או שרואים RST בלבד או סיום רגיל ב-FIN+AC.

SYN+ACK, SYN: נסתכל על שני הדגלים האלו ביחד, שביחד מהווים לנו את לחיצת היד המשלושת (כולל הACK ההתחלתי), ניתן לראות שנפתחו חיבורים תקינים ושבדפדפנים חיבורים נפתחים יותר, כנראה בשל ריבוי חלוניות.

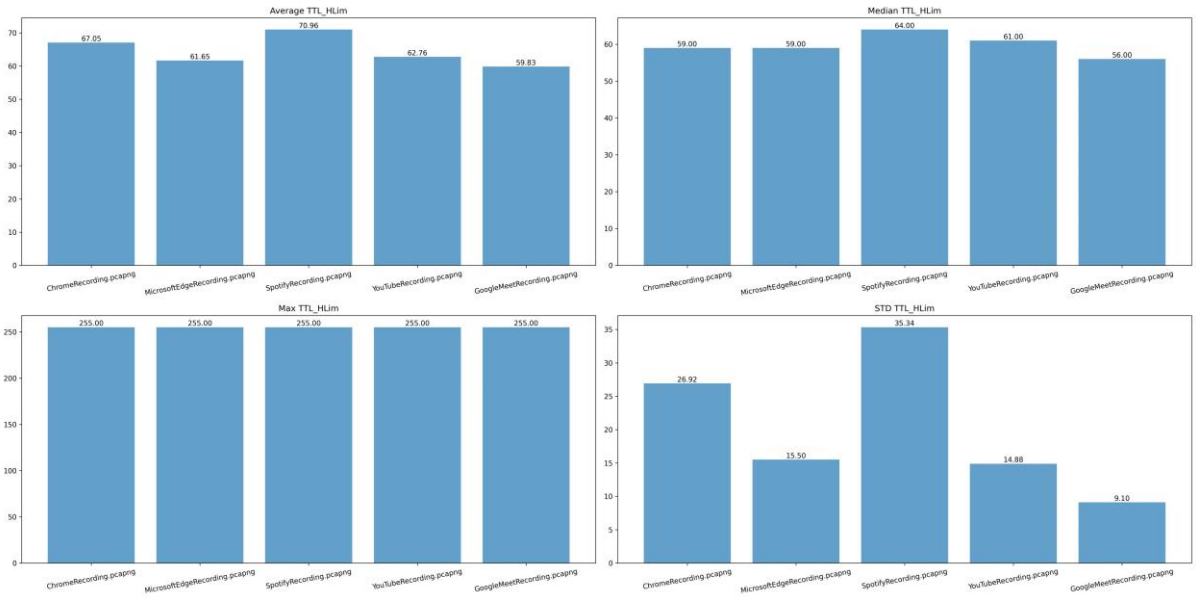
FIN+ACK: מצביע על סיום תקין של החיבור.הדפדפנים וגוגל מיט מצביעים על הרבה חיבורים שנסגרו בצורה מסודרת.מציגים מספר נמוך יחסית, דבר שעשוי להצביע על פחות חיבורים או שהסיום מתבצע חלקית בפרוטוקול אחר (QUIC במקרה של YouTube).

PSH+ACK: כמות גבוהה של PSH+ACK מעידה על יישום שדורש תגובה מיידית ללא עיכובים ניתן לראות שבגרפים המובילים אלו הדפדפנים וgoogle meet- כלומר אלו צריכים לעבור בדחיפות לצד השני, מה שהגיוני באופי זמן האמת שלהם.

YouTube משתמש כנראה בפרוטוקולים אחרים (QUIC), ולכן מופיע במספרים נמוכים יותר בתעבורת TCP הרגילה.

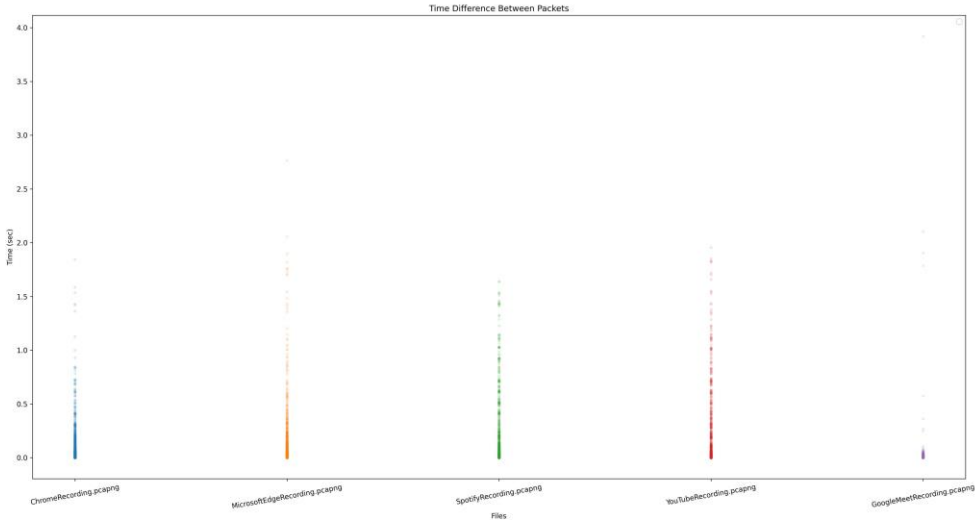
לסיכום ממבט חיצוני של תוקף לדוגמא, הוא יכול להסיק מידע רב מהדגלים ב-TCP בלי לפענח את התוכן. SYN ו-SYN+ACK חושפים פתיחת חיבורים, PSH+ACK מצביע על נתונים אינטראקטיביים (כמו שיחות או סטרימינג), FIN+ACK מסמן סיום חיבור תקין, ו-RST/RST+ACK מעיד על ניתוקים פתאומיים או חסימות. ניתוח תזמון וכמות הדגלים מאפשר לזהות סוגי שירותים ואפליקציות בשימוש, כמו דפדפן, סטרימינג או שיחת וידאו.

5. גרפים סטטיסטיים אשר מתארים את מאפיין ה- TTL עבור פרוטוקול IPV4 וכן את הhop-limit עבור פרוטוקול IPV6.



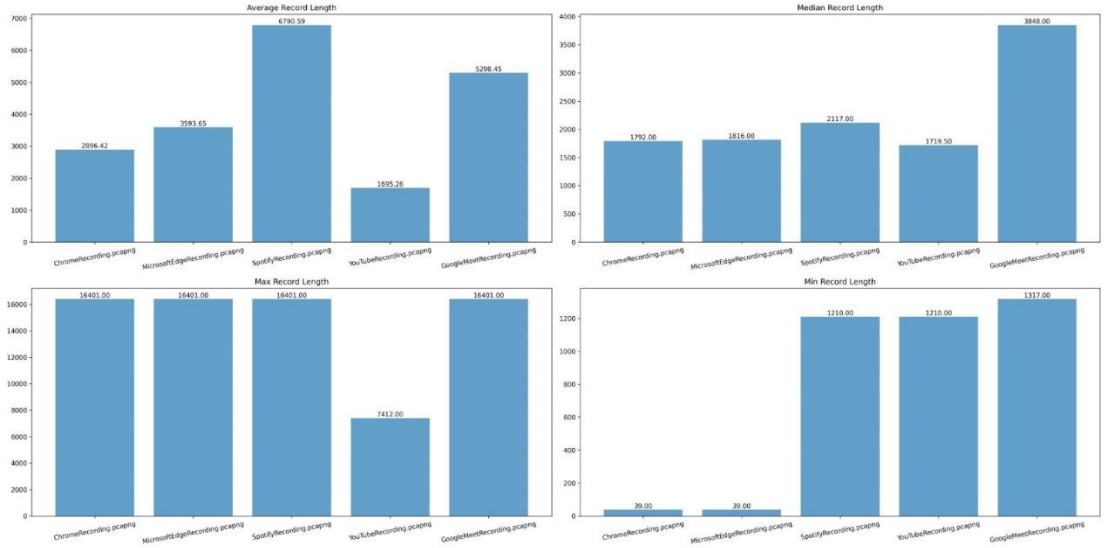
גרפים אלה מייצגים עבורנו מדדים סטטיסטיים עבור ערכי הTTL עבור IPV4 וכן Hop-Limit עבור IPV6, המדדים כוללים - ממוצע, חציון, מקסימום וכן סטיית תקן. גרפים אלה מתבטאים עבור כל אחת מההקלטות. ההבדלים העיקריים בערכים אלו יכולים לנבוע ממגוון אפשריות כגון - ההגדרה של הרשת אצלנו, מערכת ההפעלה וכמובן ממרחק התקשורת - כלומר מספר נתבים בדרך ליעד.

6. סטטיקה עבור Time Difference Between Packets -



מהגרף ניתן לראות שזמני ההמתנה בין חבילות נתונים משתנים בהתאם לקובץ . ניתן להבחין שלכל סוג תעבורה יש התפלגות ייחודית של מרווחי זמן. חלקם מציגים מרווחים צפופים יחסית עם מעט נקודות חריגות למשל Spotify בעוד אחרים מציגים פיזור רחב יותר או נקודות קיצוניות כמו Google Meet הדבר מרמז על האופי של התעבורה בכל סוג אפליקציה או דפדפן. תעבורת סטרימינג מוזיקה או וידאו עשויה לדרוש זרם נתונים רציף יחסית, ולכן הפרשי הזמן בין החבילות קטנים יותר. לעומת זאת, שיחות וידאו עשויות ליצור שינויים גדולים יותר בסדרי גודל של זמן, בגלל אופי התקשורת והדחיסה הדינמית של וידאו ואודיו בזמן אמת.

7. גרפים אשר מייצגים את מאפייני ה- TLS Header



עבור להתבונן בנתונים של TLS Header:

מאפייני הרשומות של החבילות:

עבור כל הקלטה ישנם ניתוחים סטטיסטים שאנו מפיקים, ניתוח הנתונים האלו מראים על ארבעה מדדים עיקריים שקשורים לאורך הרשומה, ערך מקסימלי, מינימלי וחציון.

נסתכל ונבין מה הממצאים:

בדפדפנים- Microsoft ו- Chrome הנתונים בכללם כמעט זהים. ניתן להבין שנשלחות רשומות בכל הגדלים- המינימלי מאוד קטן והמקסימלי גדול.

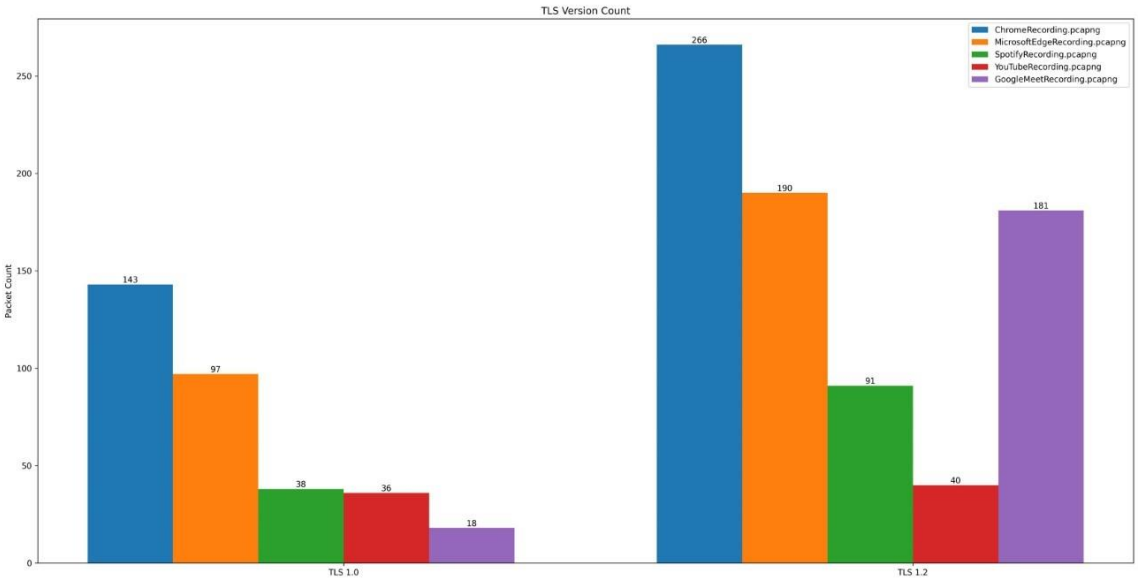
Spotify: ממוצע האורך גבוה. וגם הערכים המקסימלים והמינימלים. נוכל להסיק שהמשמעות היא שיש הרבה רשומות בינוניות וגדולות. מה שמצביע שכנראה סטרימינג מוזיקה פרקי זמן מסויים עוברים בבלוקים יחסית גדולים, כדי למנוע קפיצות בניגון.

YouTube: בניגוד לאפליקציות אחרות שמגיעות עד 16,000 בתים לרשומה, ב-YouTube אין כמעט "פיצוצים" גדולים של נתונים ב-TLS.

הסיבה היא ש-YouTube בעיקר משתמש ב-QUIC, ולכן רוב הווידאו לא נמדד כ-"TLS Record". בנתוני ה-TLS הרגילים ב-YouTube נשארות רק הבקשות הקטנות יותר (Metadata, API, תמונות ממוזערות), ולכן אורך הרשומה נמוך יחסית.

Google Meet: ניתן לראות שכל המדדים גבוהים- זה מצביע על כך שרוב הרשומות גדולות – אין כמעט רשומות קצרות כמו בדפדפנים. בשיחת וידאו בזמן אמת, האפליקציה כנראה שולחת Frames של וידאו/אודיו, בגודל די גדול, כלומר יש תעדוף לניצול רוחב הפס במלואו.

במבט מבחוץ לכל אפליקציה יש דפוס משלה של הגדלים בהם היא שולחת נתונים. יש הבדל בין דפדפנים לסטרימינגים מסוגים שונים, ככה שאפשר להבדיל.



מאפיין הTLS version:

נשווה פה בין 2 הגרסאות של TLS, גרסה 1.0 ו1.2.

ננסה להבין למה בכלל יש הבדל בגרסות ולמה יש הבדל בכמות הפאקטות הנשלחות בכל ישומון. אז אפליקציות חדשות (Google Meet) משתמשות ב-TLS 1.2 לאבטחה טובה יותר, בעוד דפדפנים תומכים גם ב-TLS 1.0 כדי לגשת לאתרים ישנים.
סטרימינג (Spotify, YouTube) פחות רגישים, ולכן עדיין משתמשים בגרסה 1.0

נראה שהדפדפנים שולחים הרבה חבילות כי הם טוענים תוכן מהרבה שרתים. יש תעדוף לגרסה 1.2 אבל גם אכן נשלחות חבילות בגרסה 1.0

YouTube/Spotify לעומת זאת שולחים פחות חבילות אבל גדולות יותר כי הם מזרימים נתונים רציפים – כי אלו אפליקציות סטרימינג – ההבדל בין כמות החבילות קיימת אך פחות בולטת.

Google Meet שולח הכי הרבה חבילות קטנות כי הוא מעביר וידאו ואודיו בזמן אמת. הוא מתעדף את גרסת 1.2 בפער גדול על 1.0 בגלל רמת האבטחה

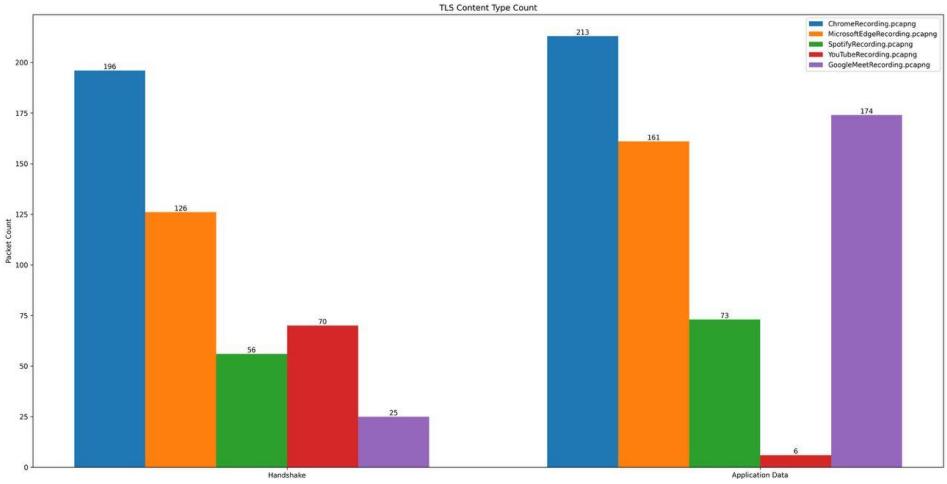
איך תוקף יכול לזהות אפליקציות?

גם אם התוכן מוצפן, תוקף יכול לזהות איזו אפליקציה בשימוש לפי כמות החבילות

דפדפנים עדיין מייצרים חבילות TLS 1.0 בכמות כי הם צריכים לתמוך באתרים ישנים.

Google Meet עובד בעיקר עם TLS 1.2 כי הוא דורש אבטחה גבוהה יותר.

סטרימינג כמו Spotify ו-YouTube משתמשים פחות ב-TLS כי הם מבוססים על פרוטוקולים אחרים לרוב.



מאפיין TLS Content Type -:

הגרפים פה מראים את יחס סוג הנתונים המועברים על פני כמות החבילות.

נסתכל על לחיצות היד ביחס למידע המועבר. באופן משמעותי chrome וMicrosoft, הדפדפנים, מקיימים לחיצת יד בכמות גבוהה מאוד. מה שהגיוני ומצביע לנו על חיבורים חוזרים ונשנים, כמו פתיחת כרטיסיות רבות או חיבורים למספר שרתים. כמו כן גם כמות גבוהה של מידע שמועבר- מידע מוצפן.

לאחר מכן Spotify- מספר בינוני של Handshake כלומר ייתכן ש-Spotify פותח כמה חיבורים במקביל (לשירותי סטרימינג, מטא-דאטה, פרסומות וכו'). עם זאת, המספר אינו גדול במיוחד מה שעשוי להצביע על כך שהזרמת המוזיקה מתבצעת בעיקר דרך חיבור אחד או מספר קטן של חיבורים. המידע שספוטיפיי מעבירה מרמזת שזה נתוני אודיו בצורה רציפה לאורך זמן.

YouTube: נשים לב שכמות לחיצות היד גבוהה לעומת מידע זעום שעובר!

הדבר מרמז שהתקשורת העיקרית של YouTube אינה עוברת ב-TLS "הקלאסי",

אלא בפרוטוקול QUIC, מה שמביא לכך שברמת ה-TLS הרגיל כמעט אין תעבורה של וידאו.

Google Meet: מעט מאוד לחיצות יד לעומת כמות מידע עצומה. מה שהגיוני כי הדבר אופייני לאפליקציה שמבצעת שיחת וידאו/אודיו בזמן אמת. נחשוב על מה קורה באפליקציות זמן אמת.

מתחברים פעם אחת (או מעט פעמים)

מעבירים כמות גדולה של נתונים מוצפנים באופן רציף (הזרמת וידאו/אודיו).

וכל זה בשמירה על TLS אחיד.

בשיחת וידאו, הגיוני להחזיק חיבור TLS אחד (או מספר מצומצם) לכל השיחה, במקום לחדש לחיצת יד. זה מפחית את העומס של חידוש חיבורים כל הזמן.

מצד התוקף – הוא עשוי לדלות נתונים כאלו כלומר על מספר הודעות Handshake לעומת כמות Application Data שהועברה – ולזהות דפוסים ייחודיים לכל אפליקציה.

Section 4- Attacker's Capabilities Discussion

הקלטנו הקלטה חדשה של שיחת ווידאו ב WhatApp הוצאנו שוב את הגרפים הרלוונטים וניסינו להשוות לאיזה סוג של תעבורה זה הכי דומה. ולהלן התוצאות :

1. <u>בהתבסס על גדלי החבילות , חותמות הזמן שלהם ומספר הזרימות</u>:

א. מספר הזרימות: בניגוד לדפדפנים כמו Chrome ו־Edge שפותחים מאות ואלפי חיבורים, WhatsApp פותח מספר של זרימות הדומה ל־Google Meet.

ב. גודל פאקטות: מתאים לתקשורת וידאו בזמן אמת, החציון והממוצע של גודל הפאקטות ב־WhatsApp גבוהים יחסית והמקסימום קרוב ל־1500 בייט. Google Meet מציג מאפיינים כמעט זהים: גודל ממוצע בסביבות 600–700 בייט, מקסימום באזור 1400–1500.

ג. הפרשי זמנים בין פאקטות: נמוכים וצפופים ב־ WhatsAppבדומה ל-GoogleMeet, הממוצע והחציון של הפרש הזמנים בין פאקטות קרובים מאוד לזה של GoogleMeet.

על סמך כל המדדים ניתן להסיק שהקלטת WhatsApp החדשה קרובה מאוד לדפוס של Google Meet, ושניהם בולטים כשיחות ווידאו.

2. <u>בהתבסס על גדלי החבילות וחותמות הזמן בלבד</u>:

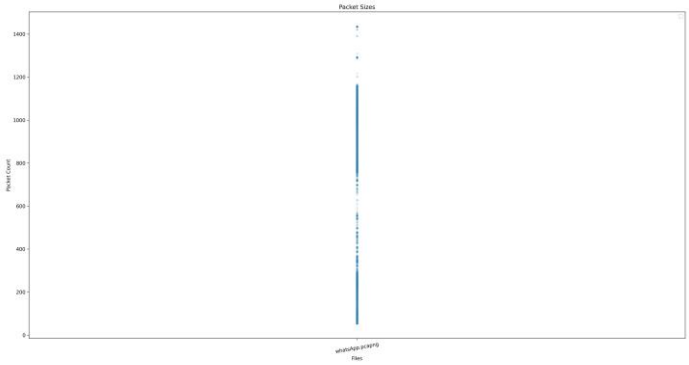
ללא מידע על מספר ה־Flows, עדיין אפשר להעריך כי ההקלטה החדשה קרובה מאוד לאפיון של Google Meet: שילוב של גדלי פאקטות ושידור בתדירות גבוהה מעיד על שיחת וידאו.

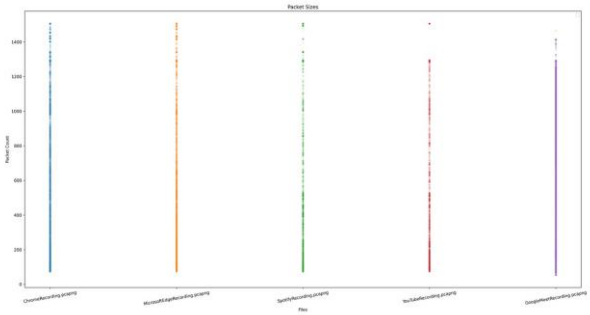
<u>סיכום</u>: בדקנו איך לזהות סוגי תעבורה בעזרת שלושה פרמטרים:

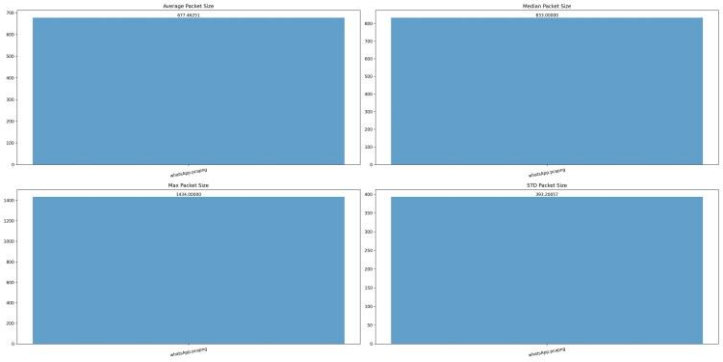
1. Flow Count – כמה זרימות נפתחו.
2. Packet Sizes – גודל הפאקטות שנשלחו.
3. Time Difference Between Packets – הזמן בין שליחת פאקטות.

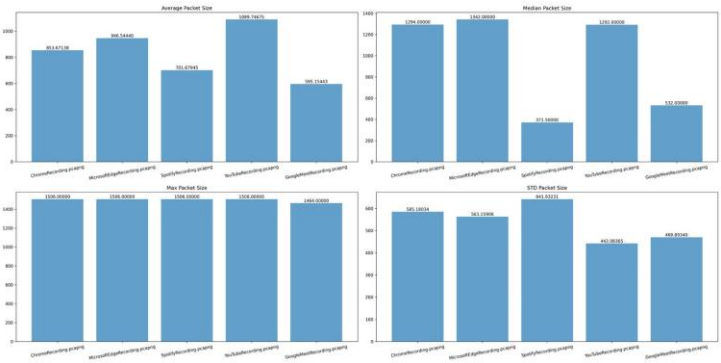
מצאנו ששיחות וידאו נראות דומה אחת לשנייה – מבחינת גודל הפאקטות, הפרשי זמנים קצרים וקצב שליחה אחיד.

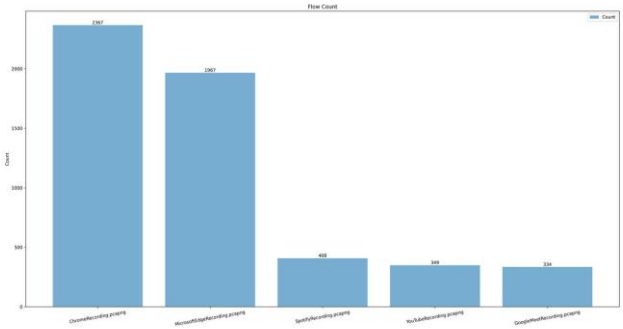
גם בלי נתוני Flow Count, אפשר עדיין לזהות שזה שיחת וידאו לפי גודל הפאקטות והקצב, אבל יותר קשה להפריד בין שירותים דומים. התמקדנו כאן רק בשיחות וידאו, וייתכן שסוגי תעבורה אחרים יהיה יותר קשה לזהות עם הפרמטרים האלה

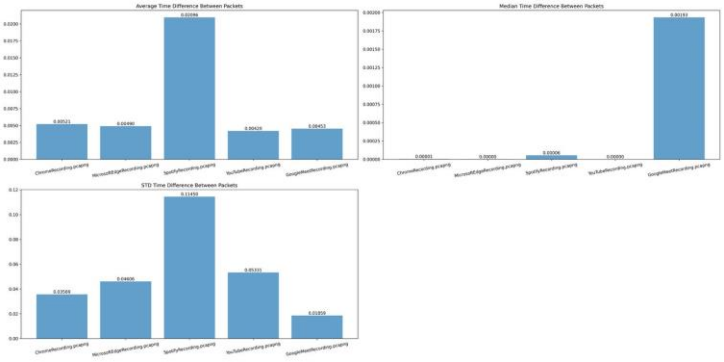


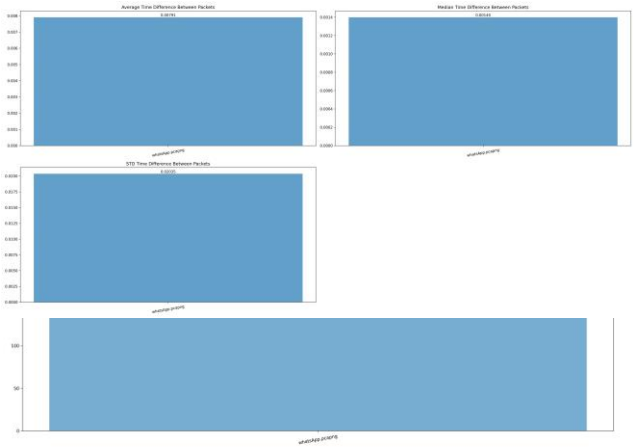


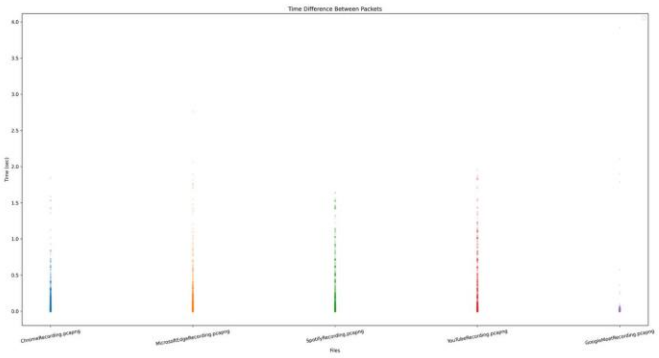


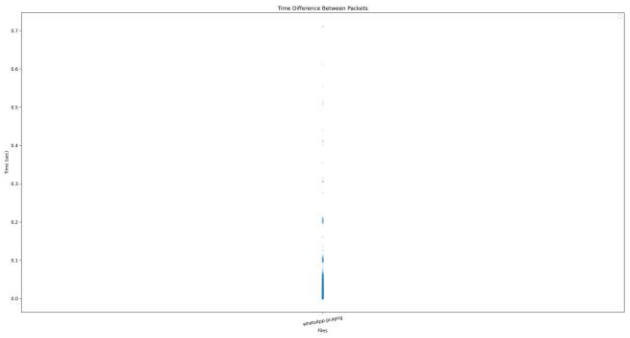





Average Time Difference Between Packets
Median Time Difference Between Packets
STD Time Difference Between Packets


Average Time Difference Between Packets
Median Time Difference Between Packets
STD Time Difference Between Packets


Time Difference Between Packets
Time (sec)
Files


Time Difference Between Packets
Time (sec)
Files

References

This part of the project was by several websites that we have found useful.

The sites are -

TLS Protocol Explanation - We had to save TSL keys, therefore we have found this useful for deeply understanding the TLS protocol.
Networking 101: Transport Layer Security (TLS) - High Performance Browser Networking (O'Reilly)

Matplotlib - This helped us with creating informative Plots.
Quick start guide — Matplotlib 3.10.1 documentation