מטלת גמר- רשתות תקשורת

מגישות: עדן חסין 207300823 , שני גולומב 325184653, מעין תורג'מן 209948058 , יובל בצר 212725048

חלק ראשון:

1. האיטיות בהעברת הקובץ יכולה להיגרם ממספר גורמים בשכבת התעבורה:

חלון קבלה קטן מדי שמגביל את כמות הנתונים מה שמאט את קצב שליחת הקובץ, עומס ברשת שגורם למנגנון הTCP להאט את קצב העברת הקובץ על מנת להתמודד עם העומס, אובדן חבילות המחייב שליחה חוזרת על פי מנגנון הTCP, זמן השהיה גבוה (Latency) עקב ריחוק פיזי בין הנתבים, חיבורים לא יציבים או לא אמינים, או הגדרות רשת שגויות כמו שגיאות בקונפיגורציה, לדוגמה הגדרה שגויה של פרוטוקולים, או מגבלות על מהירות החיבורים.

כדי לפתור זאת, נבדוק אובדן חבילות וזמן השהיה באמצעות כלים כמו Wireshark ו-traceroute, נוודא יציבות ומהירות חיבור, ונבחן שימוש ברוחב הפס ומצב השרתים. בהתאם לממצאים, נבצע אופטימיזציה של הגדרות TCP, נטפל בתשתיות הרשת או ניישם מנגנוני ניהול עומסים Quality of Service - QoS כדי להעדיף תעבורה חשובה ולשפר את זרימת המידע על מנת לשפר את מהירות ההעברה של הקובץ.

2. מנגנון ה-Flow Control בפרוטוקול TCP מווסת את קצב שליחת הנתונים כדי למנוע את הצפת המקבל בכמות נתונים גדולה מכפי שהוא יכול לעבד או לאחסן בזיכרון שלו. המנגנון מתבסס על TCP Receive Window, שמציין לשולח את כמות הנתונים שהמקבל מסוגל לקבל בכל רגע נתון.

כאשר השולח בעל כוח עיבוד גבוה בהרבה מהמקבל, עשויה להיווצר בעיית חוסר התאמה בקצב התעבורה. במצב זה, חלון הקבלה של המקבל מתמלא במהירות, מה שגורם להאטת קצב שליחת הנתונים, לעצירות זמניות ועיכובים, עד שהחלון יתפנה שוב. התוצאה היא ירידה בניצול רוחב הפס וזמן תגובה (RTT) גבוה יותר. בנוסף, התעבורה הופכת להיות לא רציפה, מה שפוגע בביצועים ובמהירות העברת הנתונים לאורך זמן.

3. במערכת ניתוב עם מספר נתיבים בין מקור ליעד, תהליך הניתוב אחראי לבחירת הנתיב האופטימלי להעברת נתונים, תוך הבטחת יעילות, אמינות וזמינות הרשת. הוא מאפשר איזון עומסים למניעת עומס יתר והפחתת זמני עיכוב, כמו גם הסתגלות לתקלות באמצעות ניתוב מחדש במקרה של כשל בנתיב והגמישות בבחירת נתיב מאפשרת לשפר את עמידות הרשת.

בחירת הנתיב משפיעה על ביצועי הרשת דרך מדדים כמו זמן השהיה, רוחב פס ואובדן חבילות. נתיב אופטימלי מבטיח מהירות גבוהה ואמינות בתקשורת, בעוד שנתיב עמוס או בעייתי עקב מרחק פיזי ארוך, עומס על נתבים, תקלות בציוד רשת או חיבור לא יציב, עלול לגרום לעיכובים ולאובדן נתונים. ההשפעה משמעותית במיוחד ביישומים רגישים לזמן אמת הדורשים זמני תגובה מינימליים.

בבחירת הנתיב יש להתחשב במדדי ניתוב כמו עיכוב, מספר קפיצות ורוחב פס זמין ואמינות הנתיב. נוסף לכך, יש לשים דגש על עומס רשת בזמן אמת ואבטחת הנתונים, כדי למנוע בעיות תקשורת ואיומים חיצוניים. שימוש בפרוטוקולי ניתוב מתקדמים מאפשר התאמה לתנאי הרשת המשתנים בזמן אמת, שיפור הביצועים הכולל וניהול יעיל של התעבורה ברשת.

4. MPTCP הוא הרחבה של פרוטוקול TCP המשפר את ביצועי הרשת על ידי שימוש במספר נתיבים להעברת נתונים בו זמנית, במקום להסתמך על נתיב בודד. יכולת זו מאפשרת שימוש במקביל בחיבורים שונים, כמו Wi-Fi ורשתות סלולריות, כדי להגדיל את רוחב הפס, להפחית את ההשהיה ולשפר את מהירות ההעברה. בנוסף, MPTCP משפר את אמינות הרשת על ידי מעבר אוטומטי לנתיב חלופי במקרה של כשל וכך מונע שיבושים בתקשורת. מנגנון איזון העומס הדינמי מעביר את התעבורה בהתאם לתנאי הרשת, מבטיח שימוש אופטימלי במשאבים ומניעת צווארי בקבוק. בשל תאימות לאחור עם TCP רגיל, כלומר היכולת של הTCP לפעול בצורה תקינה עם גרסאות ישנות יותר, MPTCP יכול להשתלב בקלות במערכות קיימות, מה שהופך אותו לפתרון אידיאלי עבור יישומים בזמן אמת, כגון שיחות וידאו ושיחות אינטרנט, ורשתות רבות חיבורים.

5. אובדן חבילות בין שני נתבים יכול להיגרם ממספר גורמים בשכבות הרשת והתעבורה. בשכבת הרשת, הגורמים העיקריים הם:

א. עומס יתר - תעבורה גבוהה מדי ברשת יכולה לגרום לעומס על הנתבים, מה שמוביל לבעיה בתהליך עיבוד החבילות וכתוצאה מכך לאובדן חבילות.

ב. בעיות ניתוב - שגיאות או חוסרים בטבלאות ניתוב או לולאות ניתוב עלולות להוביל לכך שהחבילות יבחרו נתיב לא נכון או יתקעו במעגלים, וכתוצאה מכך יאבדו.

ג. בעיות פיזיות - בעיות בכבלים, חיבורים לא תקינים או תקלות חומרה עלולות לפגוע בתקינות הרשת ולגרום לאובדן חבילות.

ד. טעויות בפרוטוקולים - בעיות בקונפיגורציות של פרוטוקולי רשת כמו IP יכולות להוביל לאובדן חבילות או חוסר תקשורת.

בשכבת התעבורה, הגורמים העיקריים הם:

א. בעיות בגודל החלון TCP - אם גודל החלון קטן מדי, יש צורך בניסיונות שליחה מחדש, מה שגורם לעומס נוסף על הרשת ויכול להוביל לאובדן חבילות.

ב. תקלות בתקשורת TCP - בעיות בקבלת הACK או במנגנון ה-flow control עשויות לגרום לאובדן חבילות.

ג. עומס יתר על חיבורים - אם יש עומס יתר על חיבורים מסוימים, פרוטוקול ה-TCP לא יוכל לעבד את כל הבקשות וזה עלול לגרום לאובדן חבילות.

כדי לפתור את הבעיה, יש לנקוט בכמה צעדים- לבצע בקרת עומס על מנת למנוע עומס יתר בנתבים, לבדוק ולוודא את תקינות טבלאות הניתוב ולעדכן אותן אם יש צורך, לוודא את תקינות החומרה והכבלים כדי למנוע בעיות פיזיות, להתאים את הגדרות ה-TCP כמו גודל חלון על מנת להימנע מעומס מיותר, להשתמש במנגנוני ניהול איכות שירות כדי להבטיח שהתעבורה הקריטית תקבל עדיפות ותמנע אובדן חבילות בתעבורת רשת חשובה.

חלק שני:

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

1. המאמר מציג תרומה משמעותית בתחום ניתוח תעבורת הרשת, עם שתי תובנות עיקריות:

א. זהו המחקר הראשון שמראה כיצד ניתן לזהות על ידי תעבורת HTTPS את מערכת ההפעלה, הדפדפן והאפליקציה שבהם משתמש הגולש, ללא צורך בגישה לתוכן ההצפנה עצמו. החוקרים פיתחו שיטה המבוססת על ניתוח דפוסי תעבורה, והוסיפו תכונות חדשות שמנתחות את אופן ההתנהגות של דפדפנים, וכן את מאפייני פרוטוקול ה-SSL. הניסויים שבוצעו הראו כי באמצעות שימוש בתכונות בסיסיות, המערכת הצליחה להגיע לדיוק של 93.51%, וכאשר שולבו גם התכונות החדשות, הדיוק עלה ל-96.06%.

ב. בנוסף, המאמר מספק מאגר נתונים רחב היקף, המכיל יותר מ-20,000 סשנים מסווגים ומתויגים, המאפשרים לחקור ולהעריך שיטות זיהוי שונות. המאגר כולל מידע על שלוש מערכות הפעלה עיקריות – Windows, Linux-Ubuntu ו-OSX, ארבעה דפדפנים מרכזיים – Chrome, Firefox Explorer ו-Safari, וכן שלוש אפליקציות פופולריות – YouTube, Facebook ו-Twitter. המאגר זמין להורדה, מה שמאפשר לחוקרים נוספים להשתמש בו לצורך מחקר ופיתוח בתחום זה.

2. המאמר משתמש בשתי קבוצות של תכונות לזיהוי מערכת ההפעלה, הדפדפן והאפליקציה של המשתמש מתעבורת HTTPS. התכונות הבסיסיות כוללות מדדים סטנדרטיים כמו מספר החבילות שנשלחו והתקבלו, כמות הבייטים הכוללת, זמני ההגעה בין חבילות, וגודל חבילות ממוצע.

התכונות החדשות, שהוצגו לראשונה במאמר, משפרות את הסיווג באמצעות שלושה סוגים עיקריים: תכונות TCP כגון גודל חלון TCP ההתחלתי, תכונות SSL כגון מספר הרחבות SSL ושיטות הצפנה, ותכונות התנהגותיות הקשורות לדפוסי "התפרצויות" תעבורה, שמאפיינות במיוחד דפדפנים, כפי שניתן לראות בגרף המוצג במאמר.

השימוש בתכונות החדשות שהוספו לראשונה במאמר זה שיפר משמעותית את דיוק הזיהוי, מ-93.51% עם התכונות הבסיסיות בלבד ל-96.06% עם שילובן של התכונות הבסיסיות ביחד עם התכונות החדשות המוצעות במאמר זה. בכך, המאמר מציג שיטה יעילה יותר למעקב אחר מאפייני השימוש במערכות הפעלה, דפדפנים ואפליקציות בתעבורה מוצפנת של HTTPS.

3. המאמר מציג מערכת סיווג שמזהה את מערכת ההפעלה, הדפדפן והאפליקציה של המשתמש מתוך ניתוח תעבורת HTTPS מוצפנת של המשתמש, תוך שימוש בתכונות בסיסיות וחדשות כפי שפרטנו בסעיף הקודם. תוצאות המחקר מראות כי שילוב התכונות החדשות שיפר את הדיוק בכל הקטגוריות, כאשר סיווג משולב (מערכת הפעלה, דפדפן ואפליקציה) הגיע לדיוק של 96.06% עם שילוב התכונות החדשות והבסיסיות לעומת 93.51% עם תכונות בסיסיות בלבד. נמצא כי מערכות הפעלה מציגות חתימות תעבורה ברורות שיחסית פשוט להבדיל ביניהן, בעוד שסיווג אפליקציות היה מאתגר יותר עקב דמיון בפרוטוקולים, ולכן חלקם סווגו בתור לא ידועים. בנוסף, ניתוח "התפרצויות" התעבורה הוכח כיעיל בזיהוי דפדפנים. התוצאות מראות כי למרות הצפנת HTTPS, ניתן להסיק מידע משמעותי על המשתמשים, דבר בעל השלכות משמעותיות על פרטיותם.

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

1. מאמר זה תורם שלושה תרומות עיקריות:

ראשית, הוא מציג מרחב נתונים חדש, גדול, מגוון וזמין לציבור עבור סיווג תעבורה מוקדם המכיל מעל 600,000 זרימות TLS הממוינות ל-19 מחלקות. מאגר נתונים זה נבנה בצורה שמשקפת את המגוון הרחב של התעבורה המודרנית והוא נאסף מאזורים גיאוגרפיים שונים. היתרון מרכזי של המאגר זה שהוא כולל שילוב של פרוטוקולי תקשורת שונים, תעבורה ממגוון שירותים פופולריים כמו סטרימינג, מדיה חברתית וגלישה באינטרנט, ותנאים רשתיים משתנים. זאת בניגוד למאגרי נתונים קודמים, שלרוב היו מצומצמים יותר ולא שיקפו בצורה טובה את האתגרים הקיימים בסיווג תעבורה מוצפנת.

שנית, הוא מציג אלגוריתם eTC חדש, ה-hybrid Random Forest Traffic Classifier (hRFTC), העולה על שיטות הסיווג הקיימות ברמת הדיוק, אשר מתבסס על שילוב בין תכונות סטטיסטיות של זרימות תעבורה לבין מאפיינים בלתי מוצפנים של TLS. האלגוריתם פותח במיוחד להתמודד עם המגבלות החדשות שמביאה הצפנת ה-ClientHello בפרוטוקול TLS 1.3 שמסתירה מידע קריטי (כמו SNI) שהיה עד כה גורם משמעותי בסיווג מוקדם של תעבורה.

שלישית, הוא מדגים את מגבלותיהם של האלגוריתמים הנוכחיים בטיפול במדגם רחב בעל הרבה נתונים והצורך באימון מחדש כאשר המיקומים הגיאוגרפיים משתנים. הבדיקות מראות כי קיימים הבדלים משמעותיים בין אזורים שונים, הנובעים משימוש בתשתיות רשת שונות, CDNים שונים ושינויים בהגדרות שרתים בהתאם לאזור הגיאוגרפי.

2. המאמר משתמש בנתונים לא מוצפנים של TLS, סדרות זמן מבוססות זרימה של התעבורה וסטטיסטיקות של גודל חבילות כתכונות סיווג עצמאיות,תוך ניצול העובדה שגם בתעבורה מוצפנת לגמרי, קיימים נתונים בלתי מוצפנים שיכולים לשמש לזיהוי וסיווג של זרימות. התכונה החדשה היא ה-hybrid Random Forest Traffic Classifier, אשר משלב את כל המאפיינים הללו במודל אחד שמשלב תכונות אלו או בראשי תיבות hRFTC.
hRFTC מתגבר על מגבלות מסווגים מבוססי חבילה או מבוססי זרימה בלבד, בכך שהוא מפיק תובנות מהנתונים הראשוניים של ה-TLS handshake וגם מדפוסי הזרימה של החבילות הבאות

אחריו. ה-hRFTC גם מרחיב את הפונקציונליות לטיפול בזרימות תעבורה של QUIC.
המאמר מציג שיטה חדשה לבחירת חבילות שמשפרת את דיוק הסיווג תוך שמירה על זמן תגובה מהיר. במקום לקבוע מראש כמה חבילות לנתח, המודל מתאים את עצמו ובוחר רק את החבילות הרלוונטיות לסיווג ומשתמש בקריטריון בחירת חבילות חדש שמשפר את איכות סיווג התעבורה תוך עמידה בדרישות השהיית סיווג תעבורה מוקדם.

3. המאמר מציג את ה-(hybrid Random Forest Traffic Classifier (hRFTC, אשר השיג שיעור דיוק של 96%, ועלה משמעותית על מסווגים קיימים. הוא מדגים יכולות הכללה חזקות, ושומר על ביצועים גבוהים אפילו עם נתוני אימון מוגבלים ועל פני מיקומים גיאוגרפיים שונים.עם זאת, נמצא כי קיימים שינויים בתצורת הרשת והתנהגות השרתים בין אזורים גיאוגרפיים שונים הנובעים בין היתר מהבדלים בתשתיות האינטרנט דבר זה מחייב אימון מחדש של המסווגים להתאמה לדפוסי תעבורה מקומיים. הניתוח מראה שגודל חבילה הוא הגורם המשפיע ביותר עבור הסיווג בנוסף גם לתכונות מבוססות זמן כמו inter-packet time יש תרומה משמעותית. בסך הכל, התוצאות מדגישות את יעילותן של גישות המשלבות בסיווג תעבורה מוצפנת את הנתונים ואת התכונות הסטטיסטיות.

FlowPic Encrypted Internet Traffic Classification is as Easy as Image Recognition

1. המאמר מציג שיטה חדשה לסיווג תעבורת אינטרנט מוצפנת באמצעות FlowPics, כלומר המרת נתוני זרימה של התעבורה לתמונות והפעלת CNN כדי לזהות קטגוריות תעבורה כגון וידאו, צ'אט ו-VoIP, וגם אפליקציות ספציפיות בדיוק גבוה 99.7%. השיטה לא דורשת הסתכלות בתוכן החבילות ולכן שומרת על פרטיות, והיא מצליחה להתמודד היטב גם עם תעבורה מוצפנת שעוברת דרך הצפנות VPN ו-Tor. בנוסף, המודל מצליח לזהות אפליקציות חדשות שלא הופיעו באימון, מה שמראה שהוא לומד דפוסים כלליים של תעבורה ולא רק מזהה אפליקציות ספציפיות. השיטה יעילה כי היא עובדת עם חלון זמן קצר וחד-כיווני, מה שהופך אותה למהירה יותר משיטות קודמות,
והתוצאות מוכיחות שהיא משיגה דיוק טוב משמעותית בהשוואה לעבודות קודמות בתחום.

2. המאמר משתמש במאפיינים מבוססי זרימה כדי לסווג תעבורת אינטרנט מוצפנת, תוך התמקדות בשני מאפיינים עיקריים: גודל החבילות, המוגבל ל-1500 בייט, וזמן הגעת החבילות, שמנורמל כך שהזמן ההתחלתי של הזרימה הוא אפס. במקום להסתמך על שיטות קודמות שמחייבות חילוץ ידני של מאפיינים סטטיסטיים או חיטוט בתוכן החבילות, המאמר מציע גישה חדשה שבה נתונים אלו מקודדים לצורה של תמונה דו-ממדית הנקראת FlowPic, בה ציר X מייצג את זמן ההגעה, ציר Y את גודל החבילות, וערך כל תא בתמונה מציין את מספר החבילות בעלות אותו גודל שהתקבלו באותו מרווח זמן.
החידוש המרכזי בגישה זו הוא בכך שהיא מאפשרת שימוש בCNN כדי ללמוד באופן אוטומטי את הדפוסים שבתמונות, בניגוד לשיטות קודמות שהצריכו עיבוד ידני של מאפיינים סטטיסטיים והשתמשו במודלים ישנים יותר.
בנוסף, השיטה שומרת על פרטיות מאחר שאין צורך לבדוק את תוכן החבילות, והיא מסוגלת להתמודד גם עם תעבורה מוצפנת VPN ו-Tor בדיוק גבוה.
בסיכומו של דבר, המאמר מציג שיטה אוטומטית ויעילה שממירה בעיה של סיווג תעבורה לבעיית זיהוי תמונה, ובכך משיגה דיוק גבוה משמעותית משיטות קודמות, תוך שמירה על פרטיות ויכולת הסקה לכלל לאפליקציות חדשות.

3. מאמר זה מציג שיטה חדשה לסיווג תעבורת אינטרנט מוצפנת בשם FlowPic. השיטה ממירה תעבורת רשת לתמונות ומשתמשת בCNN לסיווג. FlowPic השיגה תוצאות טובות בסיווג תעבורת VPN בדיוק של 98.4%, ולא בדיוק של 85% כמו שהיה בשיטות קודמות. לעומת זאת עבור Tor הדיוק היה קצת יותר נמוך 67.8%. זיהוי אפליקציות VoIP ווידאו, בוצע בדיוק של 99.7%.

FlowPic אף הצליחה לסווג אפליקציות חדשות שלא נראו קודם לכן בדיוק גבוה, מה שמראה כי הלמידה באמצעות התמונות הייתה יותר יעילה ובכך גרמה להסקה כללית לגבי מקרים שלא נלמדו.

לסיכום: FlowPic יעילה לסיווג תעבורה מוצפנת, במיוחד VPN, ומזהה אפליקציות בצורה יחסית מדויקת, אך מתקשה עם תעבורת Tor.

חלק שלישי:

סעיפים 1-3:

הקלטנו בווירשארק את התעבורה של חמשת האתרים\אפליקציות הבאות- firefox, google-chrome, spotify, youtube, zoom. על חמשת ההקלטות האלו הרצנו את הקוד (ExtractData) שכתבנו ויצרנו את הגרפים על פי השוואות שונות. השתמשנו בספריות הבאות: pandas, scapy, subprocess, collections, matplotlib, os.

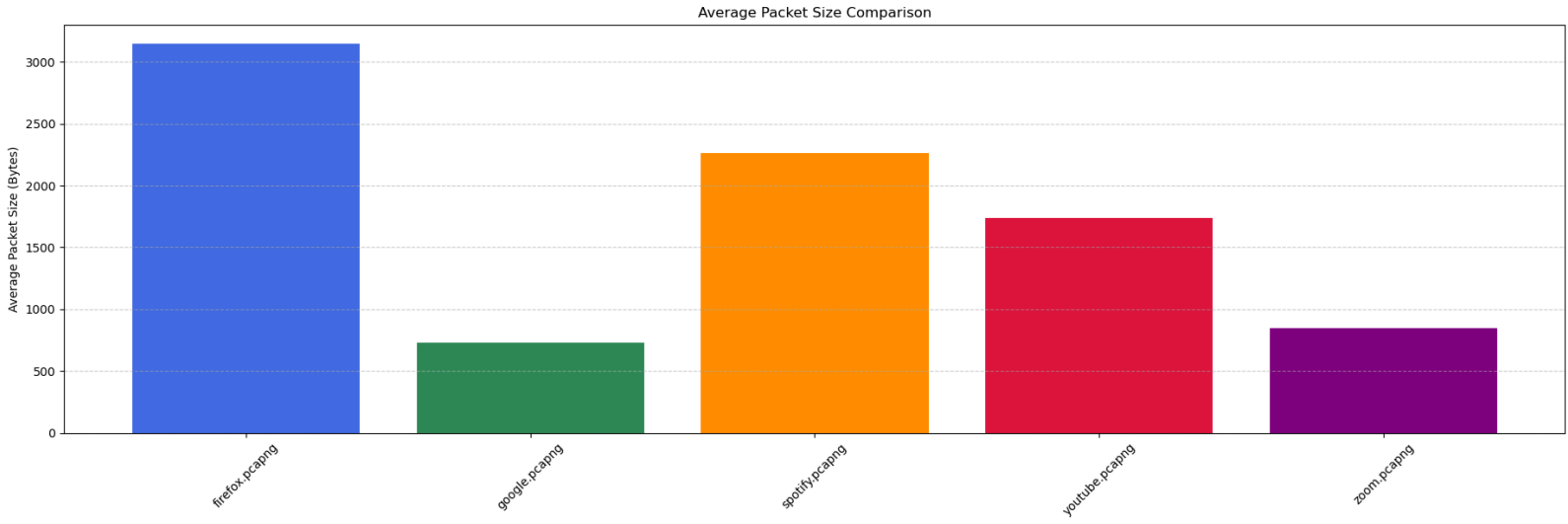
```
import os
import matplotlib.pyplot as plt
from scapy.all import IP, TCP, UDP, ICMP
from collections import Counter
import subprocess
from scapy.all import rdpcap, Raw
import pandas as pd
```

- בגרף הראשון שיצרנו ביצענו בין 5 הקלטות התעבורה השוואה על פי גודל החבילות הממוצע בכל הקלטה. חישבנו את הממוצע של גודל החבילות עבור כל הקלטה והכנסנו את כל הנתונים לגרף על מנת להציג אותם בצורה ברורה.

ניתוח הנתונים לפי הגרף:

בגרף ניתן לראות כי גודל החבילה הממוצע בהקלטה של firefox הוא הגדול ביותר לעומת גודל החבילה הממוצע בהקלטה של google-chrome שהוא הקטן ביותר.

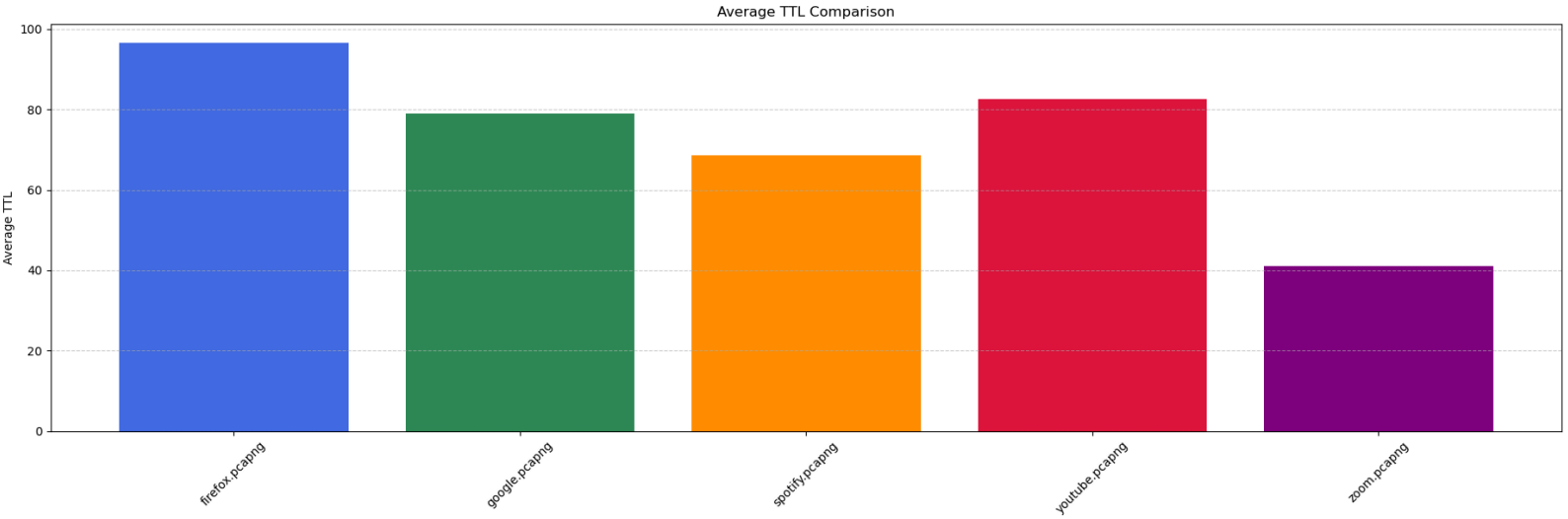
מכיון שגודל החבילה הממוצע של הקלטת תעבורת Firefox גדול מזה של google-chrome, ניתן להסיק ש-Firefox עשוי להיות מעורב בגלישה לאתרים עם תוכן כבד יותר שדורשים חבילות גדולות יותר. בנוסף, ייתכן ש-Firefox משתמש בפרוטוקולי רשת שונים או מאחסן יותר נתונים במטמון כדי לצמצם בקשות חוזרות, מה שיכול להוביל לשליחה של חבילות גדולות יותר. כמו כן, Firefox עשוי לכלול מידע נוסף כמו קבצים מצורפים או סקריפטים, בעוד ש-google-chrome עשוי להשתמש באסטרטגיות אופטימיזציה שמפחיתות את גודל החבילות.



- בגרף השני שיצרנו ביצענו בין 5 הקלטות התעבורה השוואה על פי הTTL הממוצע בכל הקלטה. חישבנו את הממוצע של הTTL עבור כל הקלטה והכנסנו את כל הנתונים לגרף על מנת להציג אותם בצורה ברורה.

<u>ניתוח הנתונים לפי הגרף:</u>

בגרף ניתן לראות כי עבור הקלטת תעבורת הfirefox גודל הTTL הממוצע היה הגדול ביותר לעומת גודל הTTL הממוצע שהיה בהקלטת התעבורה של zoom שבו הוא היה הקטן ביותר. מהגרף ניתן להסיק כי הקלטת תעבורת Firefox כוללת חבילות עם TTL גבוה יותר בהשוואה ל-Zoom, כלומר יתכן ש-Firefox משתמש ברשתות או שרתים מרוחקים יותר, שמובילים לערכים גבוהים יותר של TTL. TTL גבוה יותר יכול להעיד על רשתות מורכבות יותר או על גלישה דרך רשתות רחוקות יותר, בהן החבילות עוברות יותר קפיצות עד הגעתן ליעד. מצד שני, הקלטת הZoom עם TTL נמוך יכולה להצביע על שימוש בשרתים קרובים יותר או על חיבור ישיר יותר עם פחות קפיצות, מה שעשוי להיות תוצאה של תקשורת וידאו ישירה המבוססת על ביצועים גבוהים ומקטינה את הפיגורים והעיכובים.

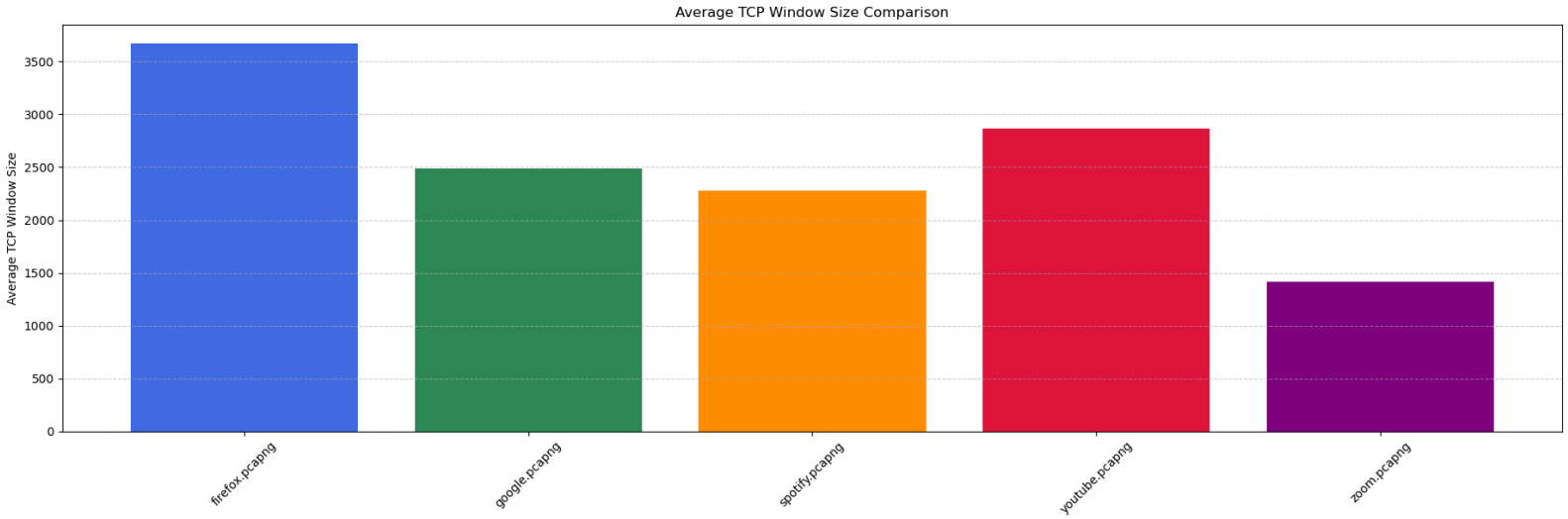


- בגרף השלישי שיצרנו ביצענו השוואה בין חמשת הקלטות התעבורה על פי גודל החלון הממוצע (בפרוטוקול הTCP). בכל הקלטה חישבנו את הממוצע של גודל החלון והכנסנו את כל הנתונים לגרף על מנת להציג אותם בצורה ברורה.

<u>ניתוח הנתונים לפי הגרף:</u>

בגרף ניתן לראות כי עבור הקלטת תעבורת הfirefox גודל החלון הממוצע הוא הגדול היותר לעומת בהקלטת התעבורה של zoom שבו גודל החלון הממוצע הוא הקטן ביותר.

מהגרף ניתן להסיק כי בהקלטת תעבורת Firefox יש שימוש בחלון TCP גדול יותר בהשוואה ל-Zoom.כלומר, יתכן כי Firefox מעביר יותר נתונים בכל חיבור TCP או מבצע יותר בקשות בו זמנית,מה שדורש חלון גדול יותר כדי לאפשר את העברת הנתונים בצורה חלקה.
לעומת זאת, הקלטת Zoom עם גודל חלון קטן יותר עשויה להעיד על תקשורת עם דרישות נמוכות יותר מבחינת רוחב פס, אולי עקב אופטימיזציות שצורכות פחות נתונים בזמן אמת- כמו תקשורת וידאו או אודיו. כלומר, Zoom כנראה מבצעת אופטימיזציה לתעבורה, במטרה להפחית עומסים על הרשת ולהבטיח איכות גבוהה של השיחה בזמן אמת, בעוד ש-Firefox עשוי להשתמש בכמות רבה יותר של נתונים.

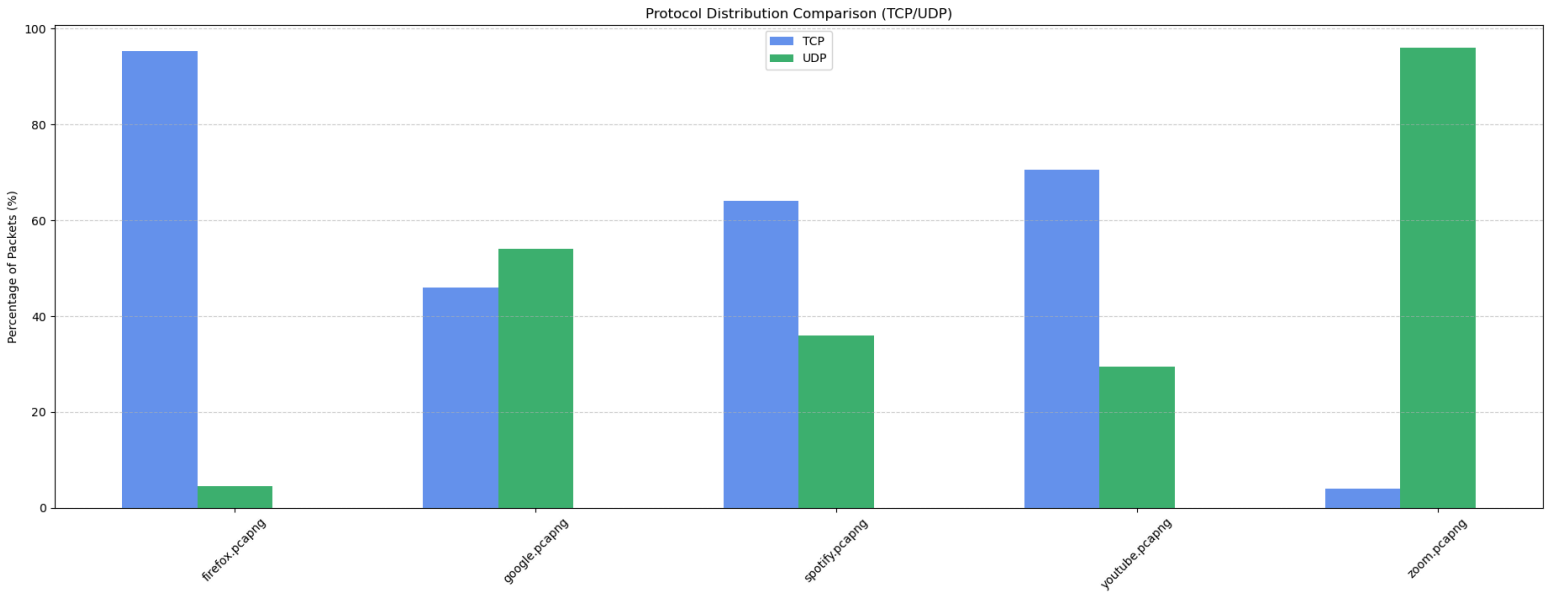


- בגרף הרביעי שיצרנו ביצענו השוואה בין חמשת הקלטות התעבורה על פי אחוז השימוש בפרוטוקול TCP לעומת פרוטוקול UDP עבור כל הקלטת תעבורה. בכל הקלטה חישבנו את אחוז השימוש בפרוטוקול TCP לעומת אחוז השימוש בפרוטוקול UDP. לאחר מכן, הכנסנו את כל הנתונים לגרף על מנת להציג אותם בצורה ברורה בכדי שנוכל לראות את ההשוואה ביניהם.

<u>ניתוח הנתונים לפי הגרף:</u>

בגרף זה ניתן לראות כי אחוז השימוש בפרוטוקול TCP בהקלטת התעבורה של firefox היא גבוהה מאוד לעומת השימוש שלו בפרוטוקול TCP. לעומת זאת, עבור הקלטת התעבורה של zoom ניתן לראות כי אחוז השימוש בפרוטוקול UDP גבוה בהרבה בהשוואה לאחוז השימוש שלו בפרוטוקול TCP.

מהגרף ניתן להסיק כי תעבורת Firefox מבוססת בעיקר על פרוטוקול TCP, שהוא פרוטוקול אמין שמתעדף שליחה סדרתית של נתונים. דבר זה מתאים לאפליקציות שבהן נדרש, אפשרות לשליחה מחדש של חבילות שאבדו, והבטחה שהנתונים יגיעו בצורה שלמה כלומר, דורשת אמינות(כמו גלישה באתרים). לעומת זאת, ב-Zoom אחוז השימוש בפרוטוקול UDP גבוה בהרבה, מה שמעיד על כך שהתעבורה שלו מבוססת יותר על UDP. פרוטוקול UDP מתאים לתקשורת בזמן אמת כמו שיחות וידאו או אודיו, שבהן חשוב להימנע מעיכובים גם אם זה אומר שניתן לוותר על שליחה מחדש של חבילות נתונים שהלכו לאיבוד. Zoom על פי הנתונים מהגרף, בוחר ב-UDP כדי להבטיח חווית תקשורת רציפה ללא השהיות, גם במחיר של פחות אמינות בהעברת הנתונים.

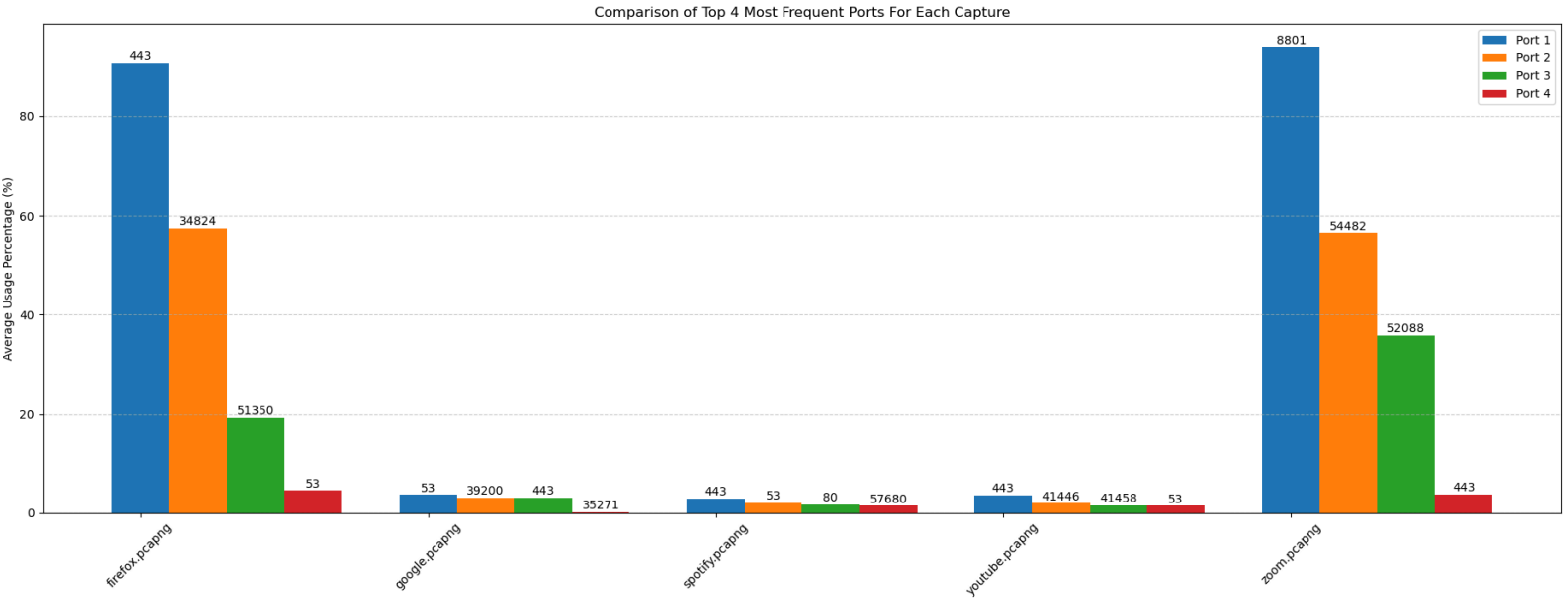


- בגרף החמישי שיצרנו ביצענו השוואה בין חמשת הקלטות התעבורה על פי ארבעת הפורטים בעלי השימוש הגבוה ביותר עבור כל הקלטת תעבורה. בכל הקלטה מצאנו את ארבעת הפורטים שבהם היה את השימוש הרב ביותר במהלך התקשורת וחישבנו את אחוז השימוש בהם. לאחר מכן, הכנסנו את כל הנתונים לגרף על מנת להציג אותם בצורה ברורה בכדי שנוכל לראות את ההשוואה ביניהם.

<u>ניתוח הנתונים לפי הגרף:</u>

בגרף ניתן לראות כי בהקלטת התעבורה של הfirefox יש שימוש רב בפורט מספר 443, המשמש לתעבורת https. לעומת זאת בהקלטת התעבורה של הZoom ניתן לראות שימוש גדול בפורטים אחרים, וגם קצת בפורט 443.

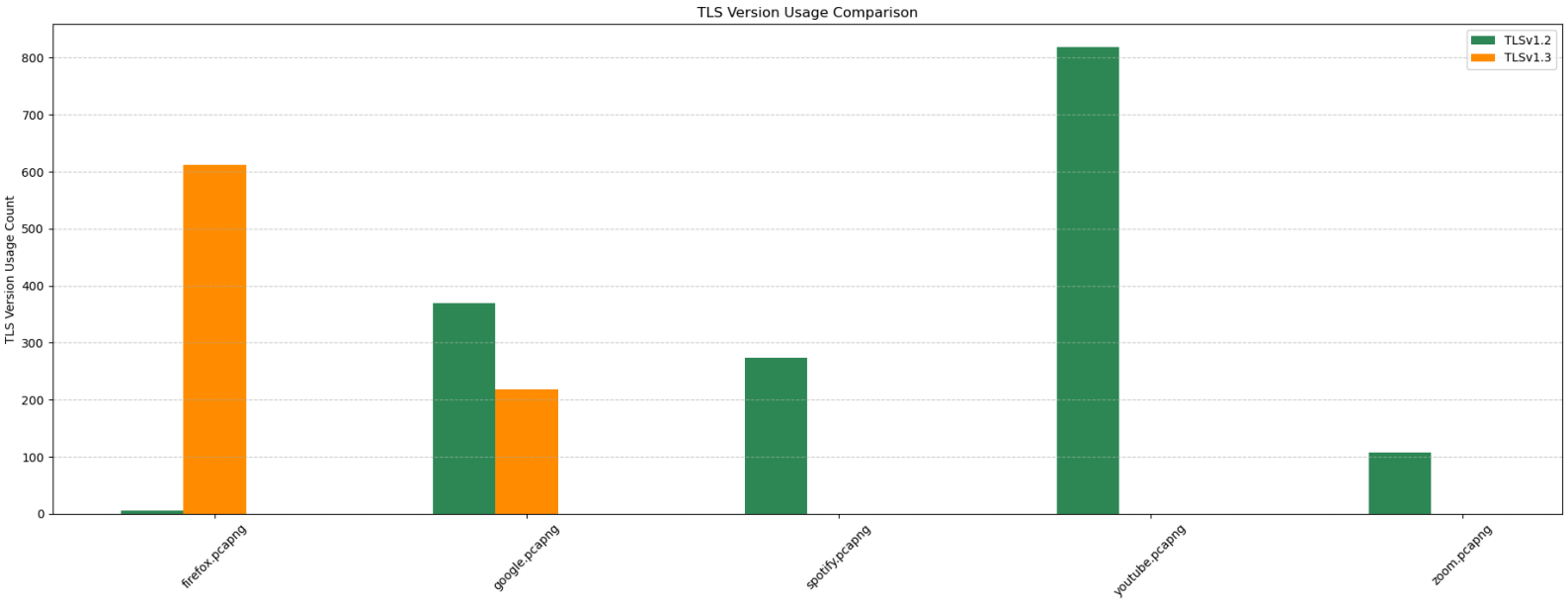
מהגרף ניתן להסיק כי תעבורת Firefox עושה שימוש נרחב בפורט 443, שהוא פורט סטנדרטי לשירותי HTTPS, מה שמעיד על כך שהגלישה באתרים בהקלטת התעבורה של Firefox כוללת הרבה פעולות שדורשות חיבור מאובטח כמו גלישה באתרים או גישה לחשבונות אישיים. לעומת זאת, בהקלטת התעבורה של Zoom ניתן לראות שימוש גדול גם בפורטים נוספים, עם אחוז קטן גם על פורט 443. זה מעיד על כך ש-Zoom עושה שימוש בפורטים שונים שאותם המחשב מקצה לשימוש, עבור תקשורת בזמן אמת, כמו פורטים נוספים לשיחות וידאו ואודיו, ולא מתמקד רק בתעבורת HTTPS כמו Firefox.



- בגרף השישי שיצרנו ביצענו השוואה בין חמשת הקלטות התעבורה על פי השימוש בגרסאות הTLS השונות (TLSv1.2, TLSv1.3) עבור כל הקלטת תעבורה. בכל הקלטה חישבנו את השימוש עבור הגרסאות TLSv1.2 ו- TLSv1.3. לאחר מכן, הכנסנו את כל הנתונים לגרף על מנת להציג אותם בצורה ברורה בכדי שנוכל לראות את ההשוואה ביניהם.

<u>ניתוח הנתונים לפי הגרף:</u>

בגרף שנוצר ניתן לראות כי בהקלטת התעבורה של הfirefox יש שימוש רב בTLSv1.3, לעומת זאת בהקלטת התעבורה של הyoutube יש שימוש רב בTLSv1.2 ואין שימוש בTLSv1.3.
מהגרף ניתן להסיק כי תעבורת ה-Firefox עושה שימוש נרחב ב-TLSv1.3, שהוא גרסה מעודכנת ומאובטחת יותר של פרוטוקול ההצפנה TLS, מה שמעיד על כך שהיישום או השירותים שמשתמשים ב-Firefox תומכים בגרסה החדשה והמומלצת מבחינה אבטחתית. לעומת זאת, תעבורת ה-YouTube עושה שימוש בעיקר ב-TLSv1.2, מה שיכול להצביע על כך שהמערכת או השרתים של YouTube עדיין לא עברו לעדכן לגרסה החדשה של TLS, או שאולי יש מגבלות מסוימות בתמיכה בגרסה החדשה בשרתים או בפרוטוקול בהתקשרות עם לקוחות.



סעיף 4:

נסתכל על שני האפשרויות:

אפשרות ראשונה - לתוקף יש את פרטי ה-Flow ID (כתובת IP מקור, כתובת IP יעד, פורט מקור, פורט יעד, פרוטוקול), גודל החבילה וחותמת זמן. עבור אפשרות זו התוקף יכול לזהות אפליקציות ואתרים על ידי:

א. ה-Flow ID מזהה את השיחה בין המחשב של המשתמש למחשב אחר באינטרנט (או שרת). הוא כולל את כתובת ה-IP של המקור והיעד, הפורטים ופרוטוקול התעבורה (TCP/UDP). התוקף יכול לעקוב אחרי השיחות בין המחשב של המשתמש לשרתים ספציפיים (כגון אתרי אינטרנט, אפליקציות, שירותי סטרימינג) ולזהות את הפעילויות של המשתמש. למשל, אם התוקף רואה הרבה חבילות עם ה-Flow ID שמקשרות לכתובת IP של שרת יוטיוב, הוא עשוי להסיק שהמשתמש גולש ביוטיוב.

ב. גודל החבילה וחותמת הזמן יכולים לעזור לזהות דפוסים אופייניים לכל אפליקציה, כמו בקשות גדולות בזמן קבוע (למשל, סטרימינג).

דרכי מניעה אפשריים:

א. הצפנת VPN או Tor: מסבכים את גילוי האפליקציה על ידי הסתרת פרטי ה-Flow ID.

ב. הסוואת תעבורה: הכנסת רעש או דחיסת חבילות בקביעות מונעת מהתוקף לזהות את סוג האפליקציה. ניתן להוסיף אקראיות או רעש לתעבורה על ידי הוספת חבילות ריקות או שינוי גודל החבילות כך שהתוקף לא יוכל לזהות דפוסים ספציפיים.

ג. הצפנת https: השימוש בפרוטוקול HTTPS (ולא HTTP) מבטיח שהמידע בין המשתמש לאתר מוצפן. גם אם התוקף יכול לראות את גודל החבילות וחותמות הזמן, הוא לא יוכל לראות את התוכן שלהן. אמנם זה לא פותר את כל הבעיות, אבל זה מונע התבוננות ישירה במידע המועבר.

אפשרות שנייה - לתוקף יש רק את גודל החבילה וחותמת הזמן. עבור אפשרות זו התוקף יכול לזהות אפליקציות ואתרים על ידי:

א. גודל החבילה וחותמת הזמן עדיין יכולים לספק מידע על האפליקציה, כי לכל אפליקציה יש דפוסי תעבורה שונים, למשל: אפליקציות סטרימינג כמו יוטיוב או נטפליקס בדרך כלל שולחות חבילות גדולות בתדירות גבוהה, במיוחד במהלך צפייה בסרטים או סדרות. אפליקציות כמו דואר אלקטרוני עשויות לשלוח חבילות קטנות יותר (למשל, הודעות קצרות או תמונות מצורפות). ככל שהתוקף יאסוף יותר מידע על הגודל והזמן שבו נשלחות החבילות, הוא יוכל לזהות את היישום שהמשתמש עושה בו שימוש.

ב. התוקף יכול להשתמש בניתוח סטטיסטי כדי לזהות דפוסים ולהסיק איזה אתר או אפליקציה משתמשת בהם. חותמת הזמן יכולה להצביע על הזמן שבו נשלחה כל חבילה, דבר שיכול לעזור לתוקף להבין את דפוס השימוש של המשתמש. לדוגמה: אם המשתמש מבצע חיבור לאתר סטרימינג בשעות מסוימות, התוקף יוכל להבחין בתדירות הגבוהה של החבילות בתקופות אלו. אפליקציות מסוימות עשויות לשלוח חבילות בתדירות גבוהה או עם צמצום בשעות מסוימות (למשל, אפליקציות הודעות או רשתות חברתיות).

<u>דרכי מניעה אפשריים:</u>

א. הסוואת גודל החבילות: ע"י הוספת אקראיות או ריפוד ניתן להסתיר את הדפוסים החוזרים. הדרך היעילה ביותר להסוות את דפוסי התעבורה היא על ידי הוספת חבילות עם גודל משתנה כדי להסוות את גודל החבילות האמיתיות. למשל, במקום לשלוח חבילה בגודל מסוים כמו 100KB, אפשר לשלוח חבילה בגודל משתנה בין 90KB ל-110KB.

ב. הצפנת VPN או Tor: מסתיר את פרטי התעבורה ומונע מהתוקף לדעת לאן נשלחו החבילות. VPN משנה את כתובת ה-IP של המשתמש ומכסה את כל תעבורת הרשת כך שהתוקף לא יכול לדעת את הכתובת הסופית של השרת שאליו שולח המשתמש את החבילות. Tor מספק שכבת פרטיות גבוהה מאוד בכך שהוא מנתב את התעבורה דרך מספר שרתים ברחבי העולם, מה שמקשה מאוד על התוקף לזהות את מקור או יעד החבילות. עם זאת, השימוש ב-Tor יכול להאט את חוויית הגלישה.

לסיכום, באפשרות הראשונה, לתוקף יש גישה לפרטים רבים יותר ולכן יכול לגלות בקלות רבה יותר באיזו אפליקציה או אתר השתמש המשתמש, גם אם התעבורה מוצפנת. באפשרות השנייה, בלי פרטי ה-Flow

ID, קשה יותר לזהות את האפליקציה, אך עדיין ניתן להסיק מידע על ידי ניתוח דפוסים של גודל החבילות וחותמות הזמן.

על מנת לבצע אוטומציה לתהליך זיהוי האפליקציות עם שני האפשרויות הנתונות בשאלה השתמשנו בלמידה בעזרת מודל הrandom forest ובספריה sklearn. כתבנו קוד (AppIdentifier) שמקבל את הקלטות הווירשארק, ממיר אותם לקבצי csv, ולבסוף מאחד אותם לקובץ csv אחד. לאחר מכן הוא מנתח את הקובץ על פי שתי האפשרויות, הראשונה- עם flowID, גודל החבילות וחתימת הזמן ועל פי האפשרות השנייה- ללא flowID אך כן עם גודל החבילות וחתימות הזמן.
בעזרת הלמידה הוא מסיק באיזה אפליקציה או אתר נעשה שימוש על ידי המשתמש, ולבסוף משווה זאת לאתר שבו המשתמש באמת ביקר.
לבסוף הצגנו את התוצאות של הלמידה, הניתוח וההשוואה לאתר\אפליקציה שבוקרו בפועל על ידי שני גרפים- אחד עבור כל אחת מהאפשרויות הנתונות בשאלה. בנוסף הקוד מפיק קובץ csv המכיל את תוצאת הניחושים אל מול האתר\אפליקציה בפועל.

הקלטנו בווירשארק את התעבורה של חמשת האתרים\אפליקציות הבאות- firefox, google-chrome, spotify, youtube, zoom. על חמשת ההקלטות האלו הרצנו את הקוד שכתבנו ויצרנו את הגרפים בעזרת למידת המכונה. השתמשנו בספריות הבאות: pyshark, pandas, numpy, os, matplotlib, sklearn.

```
import pyshark
import pandas as pd
import numpy as np
import os
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import LabelEncoder
```

נציג את התוצאות והגרפים ונפרט:

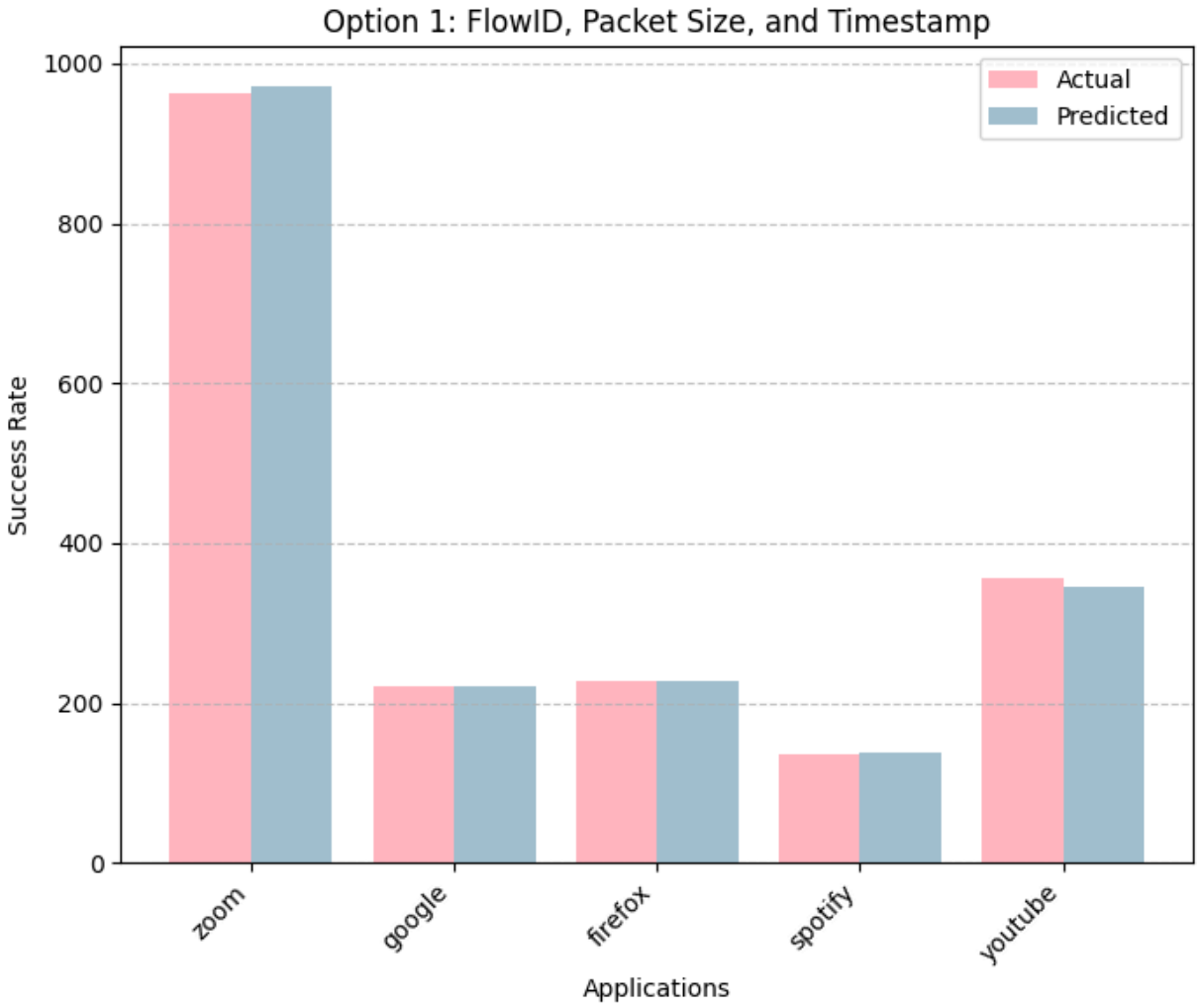
הגרף הראשון מציג את תוצאות החיזוי עבור המקרה שבו לתוקף יש את ה-FlowID, כלומר, יש לו גישה למידע על כתובות ה-IP והתעבורה בין הלקוח לשרת.

בגרף זה, לכל אפליקציה או אתר יש שתי עמודות:

1. העמודה הראשונה (בצבע מסוים) מייצגת את החיזוי (Predicted) שביצע המודל.
2. העמודה השנייה (בצבע אחר) מייצגת את המצב האמיתי (Actual), כלומר, האפליקציות והאתרים שבאמת היו בשימוש.

ניתן לראות כי יש התאמה גבוהה בין הערכים החזויים לערכים האמיתיים, כאשר אחוז הדיוק של המודל עומד על 97.90%. המשמעות היא שהשימוש ב-FlowID סיפק מידע משמעותי ששיפר מאוד את היכולת לנבא אילו אתרים ואפליקציות נפתחו בפועל.

```
Accuracy With FlowID: 97.90%
Accuracy Without FlowID: 72.07%
```



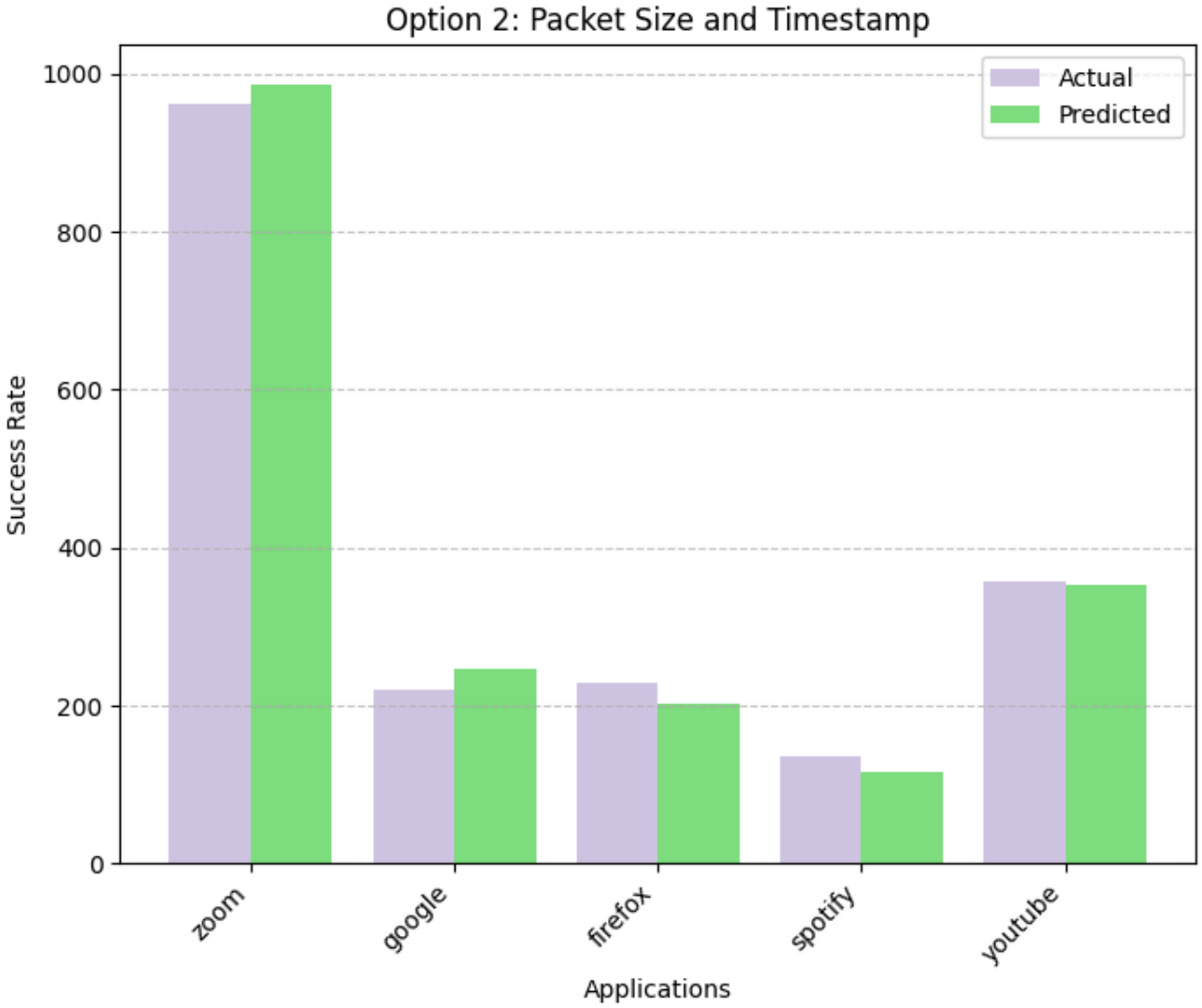
הגרף השני מציג את תוצאות החיזוי עבור המקרה שבו לתוקף אין את ה-FlowID. כלומר, אין לו גישה לכתובות ה-IP או למידע על חיבורים בין לקוח לשרת. במקרה זה, המודל נדרש להסתמך רק על חתימת הזמן וגודל החבילות כדי לנסות נבא אילו אפליקציות ואתרים נפתחו.

בדומה לגרף הראשון, גם כאן לכל אפליקציה או אתר יש שתי עמודות:

1. העמודה הראשונה (בצבע ירוק) מייצגת את החיזוי- Predicted של המודל.
2. העמודה השנייה (בצבע סגול) מייצגת את המצב האמיתי -Actual. כלומר, האפליקציות והאתרים שבאמת היו בשימוש על ידי המשתמש.

בניגוד לגרף הראשון, ניתן לראות כי אחוז ההתאמה בין החיזוי לבין המציאות נמוך יותר, עם דיוק של 72.07% בלבד. הדבר הגיוני ומתבקש, שכן כפי שהוסבר קודם, ללא FlowID, הזיהוי של האפליקציות והאתרים הופך לקשה ומורכב הרבה יותר. החוסר במידע על כתובות ה-IP וחיבורים גורם לכך שהמודל מתקשה לבצע הבחנה מדויקת בין מקורות תעבורה שונים, מה שמוביל לירידה משמעותית באחוזי ההצלחה.

```
Accuracy With FlowID: 97.90%
Accuracy Without FlowID: 72.07%
```



דוגמה לחלק מקובץ הcsv המתקבל מהרצת הקוד:

1	Length	Time_Diff	FlowID	Protocol_UDP	Actual_App	Predicted_With_FlowID	Predicted_Without_FlowID
2	1236	0.0	6	True	zoom	zoom	zoom
3	815	0.0	42	False	google	google	firefox
4	66	0.0003881454467773	44	False	firefox	firefox	firefox
5	1078	0.0005481243133544	7	True	zoom	zoom	zoom
6	66	2.193450927734375e-05	44	False	firefox	firefox	firefox
7	1132	0.0	6	True	zoom	zoom	zoom
8	11190	4.601478576660156e-05	28	False	spotify	spotify	youtube
9	86	2.193450927734375e-05	28	False	spotify	spotify	youtube
10	1330	0.0009379386901855	6	True	zoom	zoom	zoom
11	5523	0.0	1	False	firefox	firefox	firefox
12	1158	1.1920928955078125e-06	6	True	zoom	zoom	zoom
13	54218	5.698204040527344e-05	28	False	spotify	spotify	spotify
14	117	1.1920928955078125e-06	42	False	google	google	google
15	351	0.0021679401397705	16	True	zoom	zoom	zoom
16	117	0.0006730556488037	42	False	google	google	google
17	66	3.695487976074219e-05	18	False	zoom	zoom	firefox

בונוס:

באפשרות שבה יש אפליקציה עיקרית פתוחה, כמו במקרה שלנו Spotify, ואפליקציה נוספת בשימוש מדי פעם, כמו שליחת מיילים, התוקף יכול לנסות לזהות את האפליקציות או האתרים שהמשתמש ניגש אליהם על סמך דפוסים בתעבורה.

תוקף יכול לנסות לזהות איזו אפליקציה פעילה על ידי ניתוח דפוסים בתעבורה, אפילו אם התעבורה מוצפנת או מנותבת באנונימיות. כשהמשתמש מאזין למוזיקה בSpotify ומדי פעם שולח מיילים, התוקף יכול לזהות את הפעילויות על סמך שני פרמטרים עיקריים:

א. גודל חבילות התעבורה- Spotify מייצרת תעבורה רציפה וגדולה יחסית, בעוד ששליחת מיילים מייצרת תעבורה קצרה וקטנה יותר. כך תוקף יכול לזהות את דפוסי הגודל ולזהות את היישומים השונים.

ב. תזמון החבילות- כאשר Spotify פועלת, תעבורה תשלח באופן קבוע ובמרווחי זמן קבועים, בעוד ששליחת מיילים תתרחש בצורה אקראית יותר. התוקף יכול לזהות את הפעילויות על פי הזמן וההפסקות בתעבורה.

<u>דרכי מניעה אפשריים:</u>

על מנת להקשות על תוקף לזהות את הפעילויות של המשתמש, ניתן להשתמש בכמה שיטות:

א. פדינג:- הוספת פדינג או ריפוד, כלומר נתונים חסרי שימוש, לכל חבילה כך שהגודל של כל חבילה יהיה קבוע ולא יחשוף את סוג הפעילות.

ב. הסוואת זמן- הוספת עיכובים אקראיים בין החבילות או שינוי תזמון החבילות כדי להסתיר דפוסים של זמן, כמו זרימה קבועה של חבילות ממוזיקה.

ג. שימוש בפרוטוקולי הצפנה- למשל Tor, שמוסיפים אנונימיות ומקשים על תוקף לעקוב אחרי תעבורה ולזהות את היישומים.

לסיכום, שאלת הבונוס עוסקת בשאלה אם וכיצד יכול התוקף לזהות את האפליקציות השונות- Spotify ושליחת מיילים, לפי דפוסים בתעבורה, כמו גודל החבילות ותזמון השידור למרות שהן מתרחשות במקביל. על מנת להקשות על זיהוי הפעילויות, יש להשתמש בטכניקות כמו פדינג, הצפנה והסוואת זמנים.

על מנת לבצע אוטומציה לתהליך זיהוי האפליקציות כאשר שניהן קורות במקביל, השתמשנו בלמידה בעזרת מודל הrandom forest בספריות imblearn, sklearn. כתבנו קוד (Bonus) שמקבל את הקלטות הווירשארק spotify_with_gmail.pcapng, וממיר אותה לקבצי csv, על מנת שנוכל לעבוד עם הנתונים בצורה טובה.

בעזרת הלמידה הוא מסיק באיזה אפליקציה או אתר נעשה שימוש על ידי המשתמש, ולבסוף משווה זאת לאתר שבו המשתמש באמת ביקר.

לבסוף הצגנו את התוצאות של הלמידה, הניתוח וההשוואה לאתר (gmail או spotify) שבוקרו בפועל על ידי גרף. בנוסף הקוד מפיק קובץ csv המכיל את תוצאת הניחושים אל מול האתר (gmail או spotify) בפועל.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import pyshark
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report
from sklearn.preprocessing import LabelEncoder
from imblearn.over_sampling import RandomOverSampler
from ipaddress import ip_address, ip_network
import os
```

<u>נציג את הגרף שנוצר מהקוד:</u>

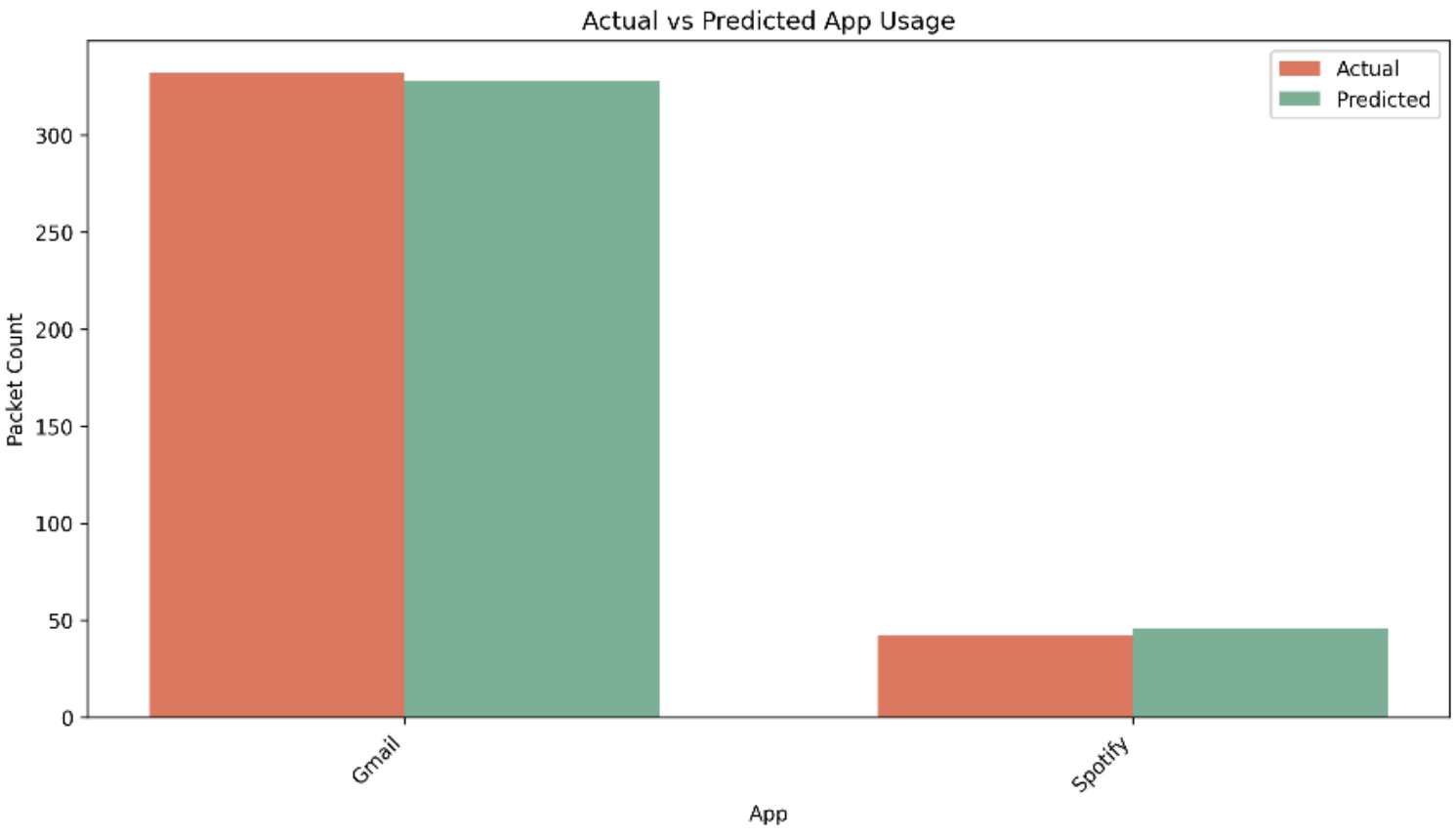
- בגרף שנוצר ניתן לראות את תוצאת החיזוי של המודל אל מול האתר שבו נעשה שימוש. בסעיף זה הקלטנו שימוש בספוטיפיי ובמקביל שלחנו מיילים. רצינו לבדוק את אחוז הצלחת המודל למרות "הערבוב" בין סוגי התעבורות ברשת המוקלטת.

לכל אתר יש שתי עמודות:

1. העמודה הראשונה (בצבע תכלת) מייצגת את החיזוי- Predicted של המודל.
2. העמודה השנייה (בצבע ורוד) מייצגת את המצב האמיתי -Actual. כלומר, האתרים שבאמת היו בשימוש על ידי המשתמש.

בנוסף ניתן לראות כי אמינות המודל היא 84.49% כלומר יחסית גבוה. לכן נוכל להסיק כי למרות שהתעבורה הכילה מספר סוגים (ספוטיפיי ושליחת מיילים) המודל הצליח לנתח זאת בצורה טובה ולחזות בצורה יחסית טובה את האתרים בהם נעשה שימוש על ידי המשתמש.

```
Model Accuracy: 84.49%
```



ביבליוגרפיה

סרטונים שנעזרנו בהם:

1. למידת מכונה + מודל Random forest

https://www.youtube.com/watch?v=_QuGM_FW9eo

https://www.youtube.com/watch?v=7eh4d6sabA0

2. יצירת plot בפייתון:

https://www.youtube.com/watch?v=X69y9N65Iu8

3. עבודה עם המפתחות:

https://www.youtube.com/watch?v=5qecyZHL-GU

ספריות בפייתון שבהן השתמשנו:

1.שאלה 3 (סעיפים 1-3): pandas, scapy, subprocess, collections, matplotlib, os

2.שאלה 3 (סעיף 4): pyshark, pandas, numpy, os, matplotlib, sklearn

3.שאלת בונוס: pyshark, pandas, numpy, os, matplotlib, sklearn, ipaddress, imblearn

שימוש בAI

.1

how can i save the keys to decrypt the traffic i capture with wireshark?

To decrypt encrypted traffic in **Wireshark**, such as **TLS/SSL**, you need to capture and save the necessary keys. The process depends on the protocol you're dealing with. Below are the main ways to save and use decryption keys.

---

1. Using SSLKEYLOGFILE (Recommended for TLS Traffic)

Many applications (e.g., browsers like Chrome and Firefox) support logging session keys. You can enable this and use the **SSLKEYLOGFILE** method in Wireshark.

Steps:

1. **Set Up Environment Variable**
   - **Windows (Command Prompt)**

   - **Windows (Command Prompt)**

     cmd Copy Edit

     ```cmd
     set SSLKEYLOGFILE=%USERPROFILE%\Desktop\sslkeys.log
     ```

   - **Windows (PowerShell)**

     powershell Copy Edit

     ```powershell
     $env:SSLKEYLOGFILE = "$env:USERPROFILE\Desktop\sslkeys.log"
     ```

   - **Linux/macOS (Terminal)**

     bash Copy Edit

     ```bash
     export SSLKEYLOGFILE=~/sslkeys.log
     ```

2. **Restart Your Browser or Application**
   - Use **Firefox**, **Chrome**, or any app that supports logging session keys.
   - Traffic will now be recorded in `sslkeys.lo`

3. **Use SSLKEYLOGFILE in Wireshark**
   - Open **Wireshark** → Go to **Edit** → **Preferences**.
   - Navigate to **Protocols** → **TLS**.
   - Under **(Pre)-Master-Secret log filename**, **browse** and select your `sslkeys.log` file.
4. **Start Capturing and Decrypt Traffic**
   - Now, encrypted **TLS** traffic should be decrypted in Wireshark.

2. בתשובה שלו היו עוד מספר ספריות שניתן לעשות בהם שימוש, אך צילמנו את העיקריות יותר.

what libraries in python can i use to make graphs?

In Python, there are several powerful libraries to create graphs and visualizations. Here are some of the most commonly used ones:

---

1. **Matplotlib** (Most Popular & Versatile)

- Great for basic and advanced plotting.
- Highly customizable.

2. **Seaborn** (Beautiful & Statistical Graphs)

- Built on **Matplotlib** with better aesthetics.
- Useful for heatmaps, violin plots, and pair plots.

5. **Pandas (Quick Built-in Graphs for DataFrames)**

- Simple visualization when working with **pandas DataFrames**.

3. גם פה היו עוד דוגמאות לספריות שלא השתמשנו בהן.

what libraries and methods in python can i use to implement the random forest model?

To implement a **Random Forest** model in Python, you can use several libraries. The most common and powerful options are:

---

1. Scikit-Learn (`sklearn.ensemble.RandomForestClassifier` / `RandomForestRegressor`)

- **Best for structured/tabular data.**
- Supports **classification** and **regression** tasks.

Installation:

bash Copy Edit

```bash
pip install scikit-learn
```

2. XGBoost (`xgboost.XGBClassifier` / `XGBRegressor`)

- **More optimized version of Random Forest (Boosted Trees).**
- **Faster & handles missing data better.**
- Supports **classification** and **regression**.

Installation:

bash Copy Edit

```bash
pip install xgboost
```

Key Methods in `RandomForestClassifier`

Method	Description
`fit(X, y)`	Train the model
`predict(X)`	Make predictions
`predict_proba(X)`	Get probabilities of each class
`score(X, y)`	Get accuracy score
`feature_importances_`	Get feature importance ranking
`set_params()`	Set hyperparameters

Would you like help tuning hyperparameters or visualizing feature importance? 🚀