**Machine Learning-Based Detection of Ransomware Using SDN**

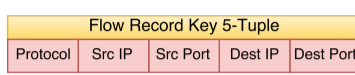
חלוקת הארכיטקטורה לשני חלקים-

#stream processing-

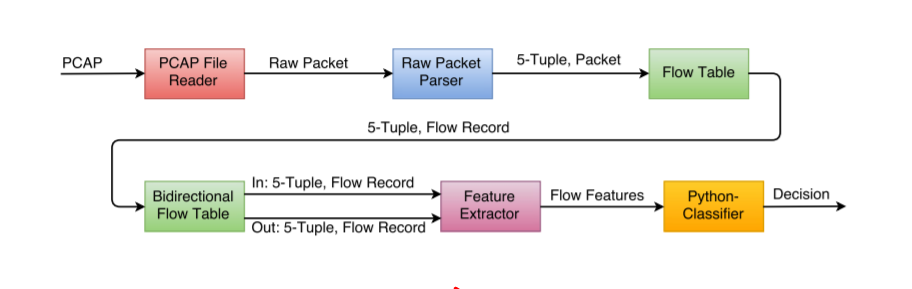
אספו רקורד של תרשימי זרימה בעזרת RaftLib’s stream processing library. (לבדוק מה זה)

בנו בעזרת RAFTLIB 5 kernelsת שעובדים במקביל וכל kernek מייצג תהליך של עיבוד המידע-

\*שימוש ב5-tuple כמפתח



מקשרים את 5 הקרנלים ולכך קרנל נותנים תפקיד-



כל פאקטה כוללת חותמת זמן, ומספר בתים, IP ןTTL ובעזרת מידע זה והקרנלים חולצו הפיצרים-

3 הקרנלים הראשונים מדממים את תהליך יצירת הflow record, מכיוון שאנו קוראים מקבצי PCAP

file reading kernel- קורא את הקבצים, ומוציא raw packet שמועברת לraw packet parser.

raw packet parser kernel- מחלץ את ה5 tuple מהפאקטה ושולח אותו עם פאקטת הפיצרים כa key-value pair אל flow table kernel.

flow table kernel- מאחסנת מפה של flow record כאשר המפתח הוא ה5-tuple והערך זה רשימה של פאקטות שנמצאות בזרימה שלה.

כאשר הקרנל הזה מקבל את ה5-tuple ופאקטת זרימה הוא מחפש בטבלה הזאת א5 tuple זהה, אם נמצא- נכסים את פאקטת הזרימה לרשימה של הפאקטות שנמצאות תחת אותו המפתח, אם לא- יוצרים כניסה חדשה בטבלה.

מכיוון שתקשורת client-sever מתחלקת לשני כיווני זרימה נצרך לבנות טבלת זרימה דו-כיוונית:

abidirectional flow table kernel- גם כאן הטבלה מנהלת רשימת זרימות אבל היא עושה התאמה בין ip וport מקור לip ןport יעד וההיפך.

ברגע שהקרנל הזה מקבל רשומה הוא מחפש את הרשומה התואמת לה בטבלה, אם נמצאה התאמה- שני הרשומות נשלחות לקרנל הבא, במידה ולא נמצאה התאמה הרשומה נכנסת לטבלה ומחכה להתאמה.

feature extraction kernel- לוקח את שתי הרשומות ומבצע חישובים על הפיצרים(ציור 2) הclassifier עושה שימוש בשני סוגי זרימה שונים- רשומות תלויות כיוון ורשומות בלתי תלויות כיוון.

זרימה בלתי תלויה- פיצרים שלא דורשים ידע של הזרימה המקבילה, והם כוללים את משך זמן הזרימה, packet interarrival times, מס כולל של פאקטות ברשימה ואת משך זמן הזרימה שלהם.

זרימת פיצרים תלויה נשענת על הידע של הזרימות שבאו בכיוון ההפוך באותו החיבור. כוללות brust length (רצף של פאקטות יוצאות שלא מכילות 2 פאקטות נכנסות סמוכות), יחס בין הפאקטות היוצאות והנכנסות ויחס של הבתים היוצאים לעומת הנכנסים.

הקרנל הזה מחשב את 2 המחלקות הללו ומעביר אותם לclassifier.

#Ransomware Classifier-

משתמשים בrandom forest ומכוונים אותו בעזרת 3 פרמטרים עיקריים:

מס העצים ביער, עומק של כל עץ ומס הפיצרים שהשתמשנו בהם בכל עץ (לפיצול)

מכייון שמטרת היעד היא להפעיל את הclassifer בline rate time אנו רוצים כמה שיותר עצים ביער בלי להוסיף תקורה משמעותית, לכן השתמשו ב40 עצים ביער וכל עץ בעל עומק של 15(הגיעו למספר הזה לאחר בדיקות שהם עשו, לא פירטו איזה)

מס הפיצרים המקסימלי יהיה השורש הריבועי של מס הפיצרים הכולל שיש בדאטה בייס, דבר זה משפר את זמן הלמידה של העץ ללא הפסד בולט בביצועי הסיווג.

חישוב הפיצרים-

