TCPSniffer

Table of Contents

[TCPSniffer 1](#_Toc47109489)

[1 מבוא 2](#_Toc47109490)

[2 Design 3](#_Toc47109491)

[2.1 High level design 3](#_Toc47109492)

[2.2 Class diagram 3](#_Toc47109493)

[2.3 Main 4](#_Toc47109494)

[2.4 Filter 4](#_Toc47109495)

[2.5 PacketData 4](#_Toc47109496)

[2.6 RawData 4](#_Toc47109497)

[2.7 PacketStructure 4](#_Toc47109498)

[2.8 Analyzer 4](#_Toc47109499)

[2.9 IpAnalyzer 4](#_Toc47109500)

[3 מדריך למשתמש 6](#_Toc47109501)

[4 Colasoft Packet builder 7](#_Toc47109502)

# מבוא

TCPSniffer מסניפה הודעות TCP ולאחר מכן מנתחת אותן כדי למצוא תקשורת חשודה.

הניתוח מתבצע עבור כתובות IP ועבור פורטים. בבסיס החישוב מחשבים את רוחב הפס ותקשורת השיר (peak) לכתובת מסוימת (ip/port) ומתבצעת בדיקה האם בזמן הדגימה האחרון של הרשת התקשורת התנהגה כבעבר או כחשודה.

המערכת מוציאה שני דו"חות אחד עבור כתובות IP חשודות ואחד עבור פורטים חשודים.

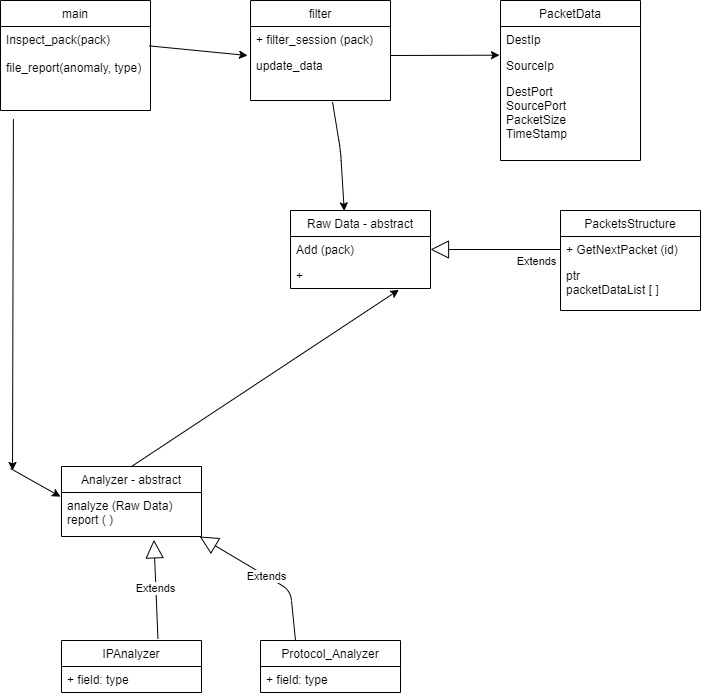
# Design

## High level design

המערכת מחולקת לשני חלקים:

1. חלק ההסנפה – מבוסס על סקפי. מסניף חבילות ה-TCP. המידע נשמר בצורה גולמי במחסן (repository).
2. חלק הניתוח – עושה שימוש בבסיס נתונים על מנת לשמור את המידע ההיסטורי (aqlite) ומקבל את הנתונים של ההסנפה האחרונה מהמחסן. מבצע את אנליזה על הנתונים כדי לגלות כתובות ip או פורטים שהתקשורת אליהם חשודה.

## Class diagram



## Main

התסריט שאחראית על יצירת האובייקטים, הפעלת הסניפר ויצירת הדו"חות.

מפעילה את הסניפר עבור כמות מסוימת של חבילות (מכונפג קקבוע).

ברגע שההסנפה מסתיימת – מפעילה את הניתוח.

בסיום הניתוח מופק דו"ח למשתמש. קובץ אחד עבור תקשורת חשודה לפורטים מסוימים וקובץ אחדש עבור תקשורת חשודה לIP מסויימים.

## Filter

במחלקה אחראית לפלטר את החבילות כך שנסניף רק חבילות TCP. היא מפרסרת את החבילות בעזרת המחלקה PacketData ובסופו של הפירסור היא שומרת עצם של PacketData במחסן לטובת שימוש עתידי של תוכנת הניתוח.

## PacketData

אחראית לפרסר חבילה מוסנפת.

## RawData

מחלקה אבסטרקטית למימוש של מחסן. מכילה אך ורק מתודה אחת – add() שמוסיפה מידע למחסן.

* בשלב זה מומש רק מחסן בזיכרון, אך בהמשך ניתן להרחיב ולממש את המחלקה בדרכים רבות.

## PacketStructure

מחלקה שיורשת מ RawData.

מממשת מחסן בזיכרון המחשב בעזרת רשימה.

המימוש של add() מוסיף כל חבילה חדשה בסוף הרשימה.

מבצעת operator overloading על מנת לממש ממשקים של iterator וכך ניתן לקבל את איברי המחסן.

## Analyzer

מחלקה אבסטרקטית של ניתוח הנתונים המוסנפים.

* בשלב זה מומש רק ניתוח לפי IP/port אך בהמשך ניתן לממש ניתוחים נוספים כגון על פי פרוטוקול.

## IpAnalyzer

מחלקה שיורשת מ-Analyzer ומממשת ניתוח של סטיות בשידור לip מסוים או פורט מסוים.

המחלקה מקבלת את הנתונים ממחסן שממש Iterator.

משתמשת ב-Sqlite על מנת לשמור את הנתונים ההיסטוריים.

בכל פעם שמופעלת פונקציית הניתוח, נקראים כל הנתונים שהוסנפו מהמחסן.

בשלב ראשון הם מקוטלגים לפי ip ולפי פורט.

בשלב שני הנתונים המוסנפים מושווים אל מול הנתונים ההיסטוריים. במידה ויש סטייה מובהקת ברוחב הפס או בכמות הנתונים שמשודרת במשך 10 שניות (פרמטר מכונפג), נרשמת רשומת התראה.

המתודה report מחזירה tuple המכיל את כל הרשומות שהוגדרו כחשודות.

# מדריך למשתמש

לפני ההפעלה הראשונה של התוכנה יש להריץ את התסריט dbCreate.py אשר בונה את בסיס הנתונים ואת הסכימה של בסיס הנתונים. בסיס הנתונים הוא SQLite והוא נשמר בקובץ test.db אשר נמצא בתיקיית הפרויקט.

לאחר שקובץ בסיס הנתונים נוצר, ניתן לגשת להפעלת התוכנה.

ההפעלה של התוכנה פשוטה –

python main.py

התוכנה תרוץ ובסיום תייצר שני קבצים:

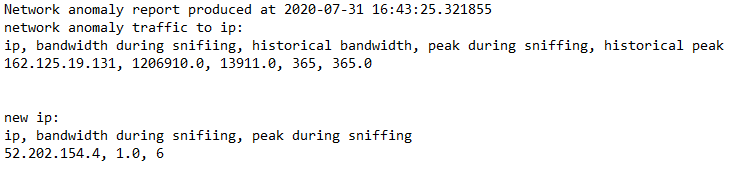
1. ip\_report.txt
2. port\_report.txt

הקבצים הם בפורמט זהה.

בראש הקובץ יש את זמן יצירת הקובץ לאחר מכן יש את רשימת ה-ip/port שרוחב הפס של התקשורת אליהם הייתה באופן מובהק גבוהה מהמוצע. הנתונים מכילים שורת כותרות ולאחר מכן נתונים בפורמט csv.

בחלק השני של הקובץ מעלי ip/port שלא היו מוכרים למערכת עד כה ושבוצעה תקשורת אליהם בהסנפה האחרונה. הנתונים מכילים שורת כותרות ולאחר מכן נתונים בפורמט csv.

דוגמה לקובץ:



# Colasoft Packet builder

על מנת לייצר תעבורת רשת ולבדוק את התוכנה נעשה שימוש בכלי Colasoft Packet builder. כלי זה נבחר לאחר מחקר מעמיק מאחר שהוא מספק ממשק גרפי מאד נוח לייצור ושידור של תקשורת נתונים.

בעזרת הכלי נבנו קבצים אשר עזרו לתוכנה לבנות את בסיס הנתונים של תקשורת סדירה וקובץ אשר בונה אנומליות ברשת ואשר התוכנה זיהתה אותה.