AI-IDS: Broad Signature Creation for Network-Exploits Via Deep learning

דו"ח סיכום פרויקט: ב'

סמסטר רישום: חורף תשפ"א

תאריך הגשה: 11/03/2021

מבצעים:

שחף הלר - Shahaf Haller

Hallershahaf@campus.technion.ac.il

עידן טאו - Idan Tau

Idan.tau@campus.technion.ac.il

מנחה:

גיל שומרון - Gil Shomron

gilsho@campus.technion.ac.il

תוכן עניינים

1. מבוא

בשנים האחרונות ישנה עלייה חדה במספר הרכיבים המחוברים לרשת האינטרנט, זאת בין היתר הודות להתפשטות טכנולוגיות IOT (Internet of Things). ע"מ לאפשר שימוש בטוח ברכיבים אלו, תוך שמירה על פרטיות ונכסי המשתמשים, עולה החשיבות של אבטחת המידע ובייחוד היכולת לזהות ולמנוע תקיפות.

כיום, משמתגלה חולשת רשת כלשהי, מתחיל מרוץ בין חברות אבטחת המידע השונות לבין האקרים ברחבי העולם. מצד אחד, חברות אבטחת המידע מנסות לאתר את מקור הפרצה, לזהות תקיפות עתידיות (בעזרת זיהוי החתימה הרשתית של התקיפה, המכונה "חתימה") ולחסום אותה. מהצד השני, מנסים גורמים זדוניים להשתמש בחולשה כמה שיותר כדי לנצל אותה עד תום. מרוץ זה הופך בשנים האחרונות למורכב אף יותר עבור חברות אבטחת המידע, ככל שיותר מהמידע שעובר ברשת הופך להיות מוצפן.

בנוסף לקשיים הקיימים בזיהוי חולשה וחסימת התקיפה, גם מהרגע שנמצאת דרך לאתר ולעצור תקיפה כלשהי, נוצרות ברשת גרסאות שונות של אותה תקיפה, אשר מנסות לעקוף את אמצעי האיתור. זאת ועוד, משגורמים זדוניים מגלים את שיטות העבודה של עמיתים, מנסים אלו לחפש חולשות נוספות סביב אותה פרצה, תחת ההנחה שבמקום בו קיימת פרצה אחת, סביר כי ישנן עוד. בפרויקט זה אנחנו מנסים לייצר "חתימות" לחולשת רשת, באמצעות רשת לומדת, זאת תוך שימוש במידע המגיע לכרטיס הרשת בלבד. בנוסף לכך, אנחנו נבדוק האם ה"חתימות" שייצרנו מצליחות לזהות גרסאות שונות של אותה תקיפה, מה שהופך אותן לכאלו שמעניקות הגנה רחבה למשתמש.

2. תקציר

כחלק מהליך המניעה של תקיפות רשת, ישנו צורך בזיהוי ה"חתימה" של התקיפה, כלומר – זיהוי הסממנים המעידים על כך שאסופת חבילות מסוימת משמשת לתקיפה. שלושה מהקשיים שניצבים בפנינו בבואנו לזהות ולמנוע תקיפות הינם:

- חלק גדול מהמידע העובר ברשת הינו מוצפן
- ישנן וריאציות רבות לאותה תקיפה
- משמתגלה פרצה המאפשרת תקיפה, נעשים ניסיונות לאיתור חולשות דומות

מטרת פרויקט זה היא בניית מערכת לומדת אשר תאפשר ייצור "חתימה" שתאפשר לזהות את התקיפה הנלמדת וכן גרסאות שונות שלה.

Abstract

As part of blocking network exploits, there is a need in identifying the exploit's "Signature", meaning – identifying the signs which indicates that a certain stream of packet is being used for exploit. Three of the main difficulties we are facing when trying to identify and prevent an exploitation from taking place are:

- A big part of networks' data is encrypted.
- There are different implementations for each exploit.
- Once a bug enabling an exploit is discovered, there is a lot of effort to find similar exploits.

3. רקע תיאורטי

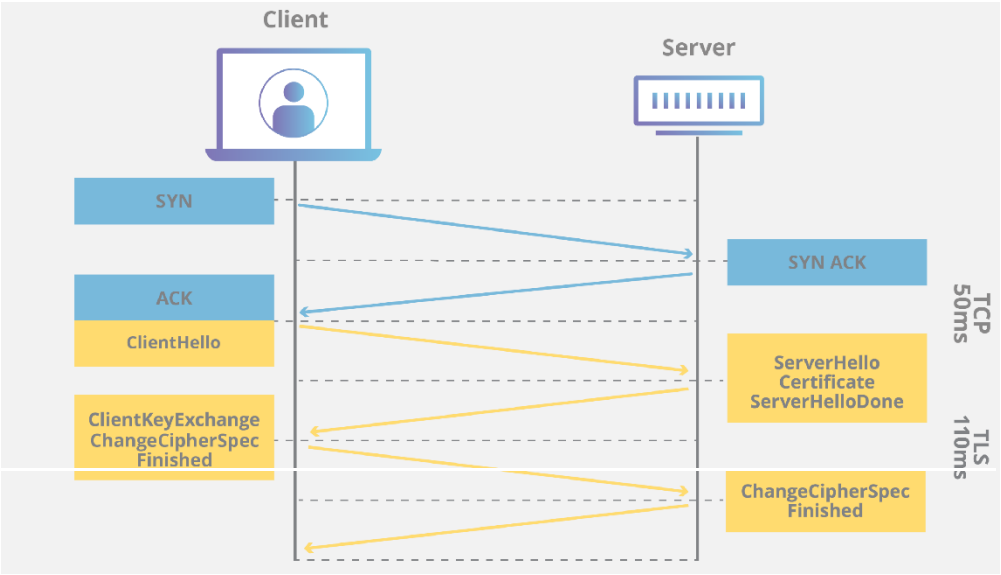
3.1. מושגים ברשתות מחשבים וסייבר

3.1.1. פרוטוקול RDP

- פרוטוקול RDP (Remote Desktop Protocol) [1] הינו פרוטוקול רב ערוצי המאפשר העברת מידע חזותי הנוגע לתצוגת המחשב, תקשורת סיריאלית של רכיב, מידע הנוגע לרשיונות, מידע פרטי רגיש (דוגמת הקלדות ותנועת עכבר) ועוד. פרוטוקול זה הינו פרוטוקול שפותח על ידי חברת Microsoft במסגרת אסופת פרוטוקולים הקרויה RDS (Remote Desktop Services) המאפשרת גישה בטוחה מרחוק לשירותים של שרת כלשהו. פרוטוקול זה יכול לעבור על גבי פרוטוקול TCP או על גבי פרוטוקול UDP כאשר בשניהם פורט ברירת מחדל הינו 3389.

3.1.2. פרוטוקול TLS

פרוטוקול TLS (Transport Layer Security) [2] [3] הינו פרוטוקול המאפשר תקשורת מאובטחת על גבי האינטרנט לאפליקציות לקוח/שרת, לרבות פרוטוקול RDP. כדי להתחיל תקשורת על גבי פרוטוקול TLS , יש לבצע מעין handshake של הפרוטוקול, המתואר באיור הבא:



איור 1 - השלבים ב-TLS handshake

3.1.3. המשמעות של Vulnerability מול Exploit

- חולשת אבטחת (Security Vulnerability) [4] הינה חולשה אותה גורם עוין יכול לנצל כדי לסכן את החיסיון, הנגישות או את האמינות של מקור. ניתן להסתכל על חולשה זאת כבאג (לוגי, טכנולוגי או אחר) אשר ניתן לנצלו כדי לחבל בשימוש המקורי לא יועד האמצעי בו קיימת החולשה.
- חולשה / ניצול חולשה (Exploit) [4] הינה מקטע קוד אשר עוצב במיוחד על מנת לנצל חולשת אבטחה בכדי במשאב כלשהו שלא למטרה שלשמה הוא נועד.

3.1.4. השמשות שונות של חולשה

השמשה (Implementation) של חולשה הינה למעשה גרסה מסוימת של ניצול חולשה. ניתן להשתמש בקטעי קוד שונים, המובילים להתנהגויות של הצד היוזם את ניצול החולשה, אשר בכולן משתמשים באותה שיטה כללית של ניצול חולשה ובעצם מנצלים את אותה חולשת אבטחה.

3.1.5. מערכת לגילוי חדירות – IDS (Intrusion Detection System)

מערכת לגילוי חדירות [5] הינן מערכות שיעודן לזהות מצבים בהם נעשה ניסיון לפגוע בהתנהלות התקינה של משאבים שונים ברשת מחשבים. ישנם שני סוגים עיקריים של מערכות שכאלו:

- HIDS (Host-Based IDS) – מערכות אשר עובדות על שרתים ומחשבי קצה וייעודן לזהות איומים המגיעים מבחוץ (דרך תעבורת הרשת) או כאלו הקיימים בפנים (על ידי ניטור תהליכים פנימיים וכו'). החיסרון המרכזי של מערכות אלו הוא ביכולתן למנוע תקיפות טרם הגעתן על הרכיבים עליהם הוא מצוי.
- NIDS (Network-Based IDS) – מערכות אשר מנסות להגן על כלל הרשת, על ידי ניטור התעבורה ברשת (בין אם בנקודה אחת ובין אם על ידי ניתוח כלל תעבורת הרשת).
  החיסרון המרכזי של מערכות אלו הוא בחוסר יכולתן לנטר את הנעשה ברכיבי הקצה ברשת (בדומה ל HIDS).

3.1.6. חתימות (Signatures)

היסטורית, חתימות [6] [7] הינן רצף בתים אשר מזוהה עם רושעה מסוימת. משמעות הדבר היא שרצף הבתים הזה נפוץ בקוד של הרושעה או של הקבצים שהרושעה מייצרת אף לא בקבצים ותוכנות אחרות. כיום ישנם סוגים נוספים של "חתימות" אותם ניתן לחלק לקטגוריות הבאות:

- חתימה סטאטית – חתימה אשר, בדומה לתיאור ההיסטורי, מזהה רצף בתים אשר נפוץ בקוד של רושעה או ניצול חולשה כלשהם או בקבצים שהרושעה מייצרת
- חתימה דינאמיות-התנהגותיות – תיאור מפורט של סדר הפעולות בה נוקטת רושעה או אשר מבוצעות במסגרת ניצול חולשה כלשהי. בעזרת התיאור, ניתן לזהות במהלך הריצה (בין אם בסביבה מבוקרת ובין אם לאו) האם מדובר בקטע קוד זדוני מוכר.
- חתימה היוריסטית – תיאור חלקי של התנהגויות המאפיינות רושעות וניצולי חולשות למיניהם.

בעזרת תיאורים אלו ניתן לאתר רושעות וניצולי חולשות שעדיין לא היו מוכרים או לחלופין לזהות מימושים חדשים של רושעות וניצולי חולשות מוכרים.

3.1.7. חולשת Bluekeep

חולשת Bluekeep (CVE-2019-0708) [8] [9] הינה מימוש חולשה במנגנון RDP, האפשרית בשל באג באחד הדרייבים המתמודדים עם הפרוטוקול. מאפשר למשתמש זדוני, בלא הזנת פרמטרי התחברות, להריץ קוד על רכיב יעד עליו קיים RDP client, זאת על ידי שליחת חבילות מידע אשר עוצבו מראש. בהפשטה, השיטה בה משתמשת החולשה, הינה להגדיר כחלק מפרוטוקול RDP, אשר בו ניתן להגדיר ערוצים וירטואלים לחיבור, ערוץ כפול, הזהה לערוץ מסוים אשר קיים כבר, ודרכו להריץ קוד במיקום כלשהו בזיכרון של רכיב היעד. על בסיס חולשה זאת, נכתבו השמשות אשר מאפשרות לאתר את המיקום בזיכרון בו רץ הקוד ובעזרת טכניקות נוספות, אף לאפשר הרצת כל קוד שהמשתמש הזדוני רוצה על רכיב היעד.

3.1.8. פרויקט Metasploit

פרויקט מטהספלויט (Project Metasploit) [10] הוא כלי המיועד למבדקי חדירה (Penetration testing) ומכיל בתוכו מאגר נתונים והשמשות לחולשות נגד מערכות הפעלה, מערכות אנטי וירוס ותוכנות שונות.

לפרויקט יש ממשק בשם Metasploit Framework Edition המכיל ממשק פקודה, יבוא תוספות, ביצוע תקיפה כוחנית (Brute force) באופן ידני, סריקת פורטים, מהדר עבור שפת Ruby ועוד.

לרוב, השימוש במאגר בכלל ובממשק בפרק יכונה שימוש ב Metaploit.

3.2. מושגים במערכות לומדות

3.2.1. מערכות לומדות

המושג מערכות לומדות [11] מתייחס לזיהוי האוטומטי של דפוסים משמעותיים במידע.

בעשורים האחרונים תחום זה הפך להיות כלי משמעותי בכמעט כל משימה שדורשת חילוץ תובנות ממאגרי מידע. אחד מהפיצ'רים הנפוצים בהם נעשה שימוש בתחום זה, בניגוד לשימוש המסורתי במחשבים, נובע מכך שמורכבות הדפוסים אותם יש לזהות, מונעים מהמתכנת האנושי לספק תיאור מפורש, מדויק ומפורט של אופן ביצוע המשימה. מערכות לומדות עוסקות בהקניית יכולת הלמידה וההתאמה של תוכנות למידע הנלמד.

3.2.2. רשת נוירונים

רשת נוירונים [11] מלאכותית הינה מודל חישובי אשר שואב השראה מהמבנה של רשתות הנוירונים במוח. אם נפשט את מבנה המוח, הוא מכיל מספר גדול של יחידות חישוב בסיסיות (נוירונים) המחוברות זו לזאת ברשת תקשורת מורכבת, אשר באמצעותה המוח מסוגל לבצע חישובים מורכבים.

ניתן לתאר רשת נוירונים כגרף מכוון אשר הצמתים שלו הינם הנוירונים והקשתות הינן החיבורים בין הנוירונים. כל נוירון מקבל כקלט סכום ממושקל של מוצאי הנוירונים המחוברים אליו דרך קשתות נכנסות.

רשתות נוירונים נבדלות זו מזאת במשקלי הנוירונים, מבנה שכבות הנוירונים, כמות השכבות, כמות החיבורים בין הנוירונים ואף כמות הנוירונים עצמה.

3.2.3. שכבות נפוצות ברשתות נוירונים

ברשתות נוירונים, ישנו שימוש בשכבות נוירונים [12], כאשר כל שכבה מכילה פונקציה שונה על הקלט המגיע אליה, טרם העברת המידע אל השכבה הבאה. מבין השכבות הנפוצות בהן נעשה קיימות השכבות הבאות:

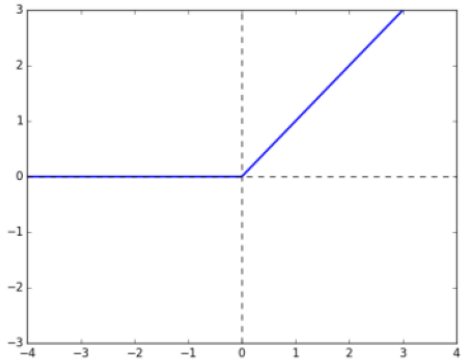
- שכבת קונבולוציה – כל איבר במוצא הוא מכפלה של כל איברי הקלט בסט המשקולות של השכבה. כלומר, כל איבר במוצא הינו קומבינציה לינארית של איברי הכניסה.

  הנוסחה הכללית של השכבה הינה:

$$(f * g)[n] = \sum_{i=0}^{m} \sum_{j=0}^{k} f[i]g[i+j]$$

- ReLU (Rectified Linear Unit) – שכבה בה כל איבר במוצא הינו המקסימום בין ערכי הקלט לבין 0. את הפונקציה הנ"ל ניתן לתאר באופן הבא:

$$ReLU(x) = (0, x)$$



איור 2 - תיאור גרפי של שכבת Relu

- שכבת maxpool – לשכבה זו מגדירים רוחב גרעין וצעד ועבור כל צעד המוצא הוא הערך המקסימלי אשר נמצא בתוך רוחב הגרעין.

איור 3 - תיאור גרפי של שכבת maxpool

- שכבה לינארית – שכבה זאת מייצרת סכום ממושקל של ערכי הכניסה ובכך מאפשרת להמיר קלט ברוחב מסוים, לפלט בעל רוחב אחר. הפונקציה הכללית אותה מממשת השכבה הינה:

$$f(x_i) = \sum_{d=1}^{D} w_d x_i^d + b = \sum_{d=1}^{D+1} w_d x_i^d$$

כאשר:

  - w = וקטור המשקלים
  - b = פרמטר הטיה

3.2.4. <u>פונקציית הפסד</u>

פונקציית הפסד [11] מוגדרת כך: בהינתן קבוצה H אשר מהווה את ההיפותזה או המודל שלנו ותחום Z ותהי פונ' כלשהי $H \times Z$ על מרחב המספרים הממשיים האי-שליליים, אזי פונקציית ההפסד הינה:

$$l: H \times Z \to R_+$$

כפי שציינו לעיל, מטרת המערכת הלומדת הינה לזהות דפוסים מורכבים מתוך מידע, זאת בשיטות של למידה והתאמה. מטרת פונקציית ההפסד היא לאמוד את איכות זיהוי הדפוסים של המערכת (הנקרא חיזוי), זאת ביחס לדפוסים אשר זוהו ואומתו זה מכבר. השאיפה הכללית בבניית מערכות לומדות הינן להביא את פונקציית ההפסד למינימום.

3.2.5. <u>תהליך האופטימיזציה</u>

תפקיד האופטימיזציה הינו מציאת נקודת המינימום של פונקציית ההפסד. שיטת משמעותית בה נעשה שימוש בשיטות אופטימיזציה רבות למציאת נקודת המינימום הינה אלגוריתם גרדיאנט יורד.

נתאר כעת את האלגוריתם הנ"ל וכן שני אלגוריתמי אופטימיזציה העושים שימוש בו:

3.2.5.1. גרדיאנט יורד ***<u>(Gradient Descent)</u>***

הגרדיאנט של פונקציה גזירה $f: R^d \to R$ ב w, הינה וקטור הנגזרות החלקיות של:f

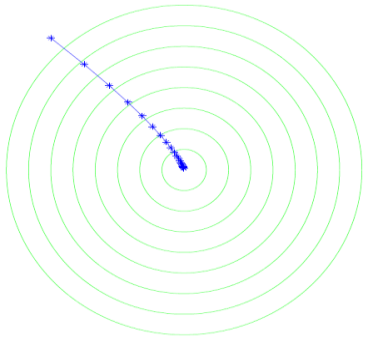
$$\nabla f(w) = \left(\frac{\partial f(w)}{\partial w[1]}, \dots, \frac{\partial f(w)}{\partial w[d]}\right)$$

אלגוריתם הגרדיאנט היורד [11] הינו אלגוריתם איטרטיבי. מתחילים בערך התחלתי .w לאחר מכן, בכל איטרציה, אנחנו עושים צעד בכיוון של הגרדיאנט השלילי בנקודה. כלומר, צעד העדכון הינו:

$$w^{(t+1)} = w^{(t)} - \eta \nabla f\left(w^{(t)}\right)$$

כאשר $\eta > 0$ הינו סקלר הקרוי קצב הלמידה.

לבסוף, לאחר T איטרציות, במוצא האלגוריתם יתקבל הוקטור הממוצע: $\underline{w} = \frac{1}{T}\sum_{t=1}^{T} w^t$.



איור 4 - תיאור ויזואלי של אלגוריתם הגרדיאנט היורד

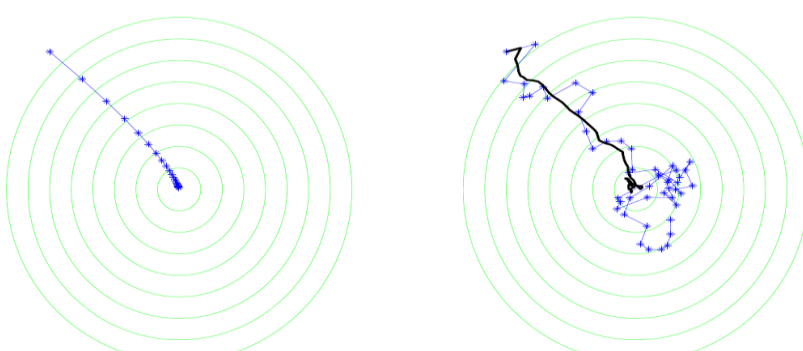
3.2.5.2. *אלגוריתם* ***SGD (Stochastic Gradient Descent)***

באלגוריתם הגרדיאנט הסטוכסטי היורד [11], איננו דורשים לעדכן את הכיוון בדיוק על פי הגרדיאנט. במקום זאת, אנו מאפשרים לכיוון להיות וקטור רנדומאלי ודורשים שהערך הצפוי בכל איטרציה תהיה שווה לכיוון הגרדיאנט.

ניתן לתאר את אלגוריתם הגרדיאנט הסטוכסטי היורד באופן הבא:

Stochastic Gradient Descent (SGD) for minimizing $f(\mathbf{w})$

parameters: Scalar $\eta > 0$, integer $T > 0$
initialize: $\mathbf{w}^{(1)} = \mathbf{0}$
for $t = 1, 2, \ldots, T$
  choose $\mathbf{v}_t$ at random from a distribution such that $\mathbb{E}[\mathbf{v}_t \,|\, \mathbf{w}^{(t)}] \in \partial f(\mathbf{w}^{(t)})$
  update $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta \mathbf{v}_t$
output $\bar{\mathbf{w}} = \frac{1}{T}\sum_{t=1}^{T} \mathbf{w}^{(t)}$



איור 5 - תיאור ויזואלי של אלגוריתם SGD

3.2.5.3. ***אלגוריתם RMSprop***

אלגוריתם RMSprop [11] הינו אלגוריתם ההמבוסס על אלגוריתם הגרדיאנט היורד, בעל קצב למידה המתאים

את עצמו. נסמן ב g את הגרדיאנט הדועך של הפונקציה ונקבל כי הנוסחה לאלגוריתם זה הינה:

$$E[g^2]_{(t+1)} = 0.9E[g^2]_{(t)} + 0.1g_{(t)}^2$$

כאשר:

$$w^{(t+1)} = w^{(t)} - \frac{\eta}{\sqrt{E[g^2]_{(t)} + \varepsilon}} g_{(t)}$$

נציין כי באופן כללי ממוצע השורש הריבועי (root mean squared – RMS) מוגדר כ:

$$RMS\left(g_{(t)}\right) = \sqrt{E[g^2]_{(t)} + \varepsilon}$$

- מטרת ה-ε היא למנוע התאפסות של ה RMS.

3.2.6. <u>מאגרי מידע</u>

מאגר המידע (Data set) [12] עליו עובדת המערכת. מאגר מידע זה צריך להכיל את כלל הפרמטרים מהם מצפים שהמערכת תלמד.

3.2.7. <u>סדרות אימון וולידציה</u>

על מנת לאמן את המערכת, מזינים אליהם מאגר נתונים מקוטלג, כאשר עבור כל דוגמית מידע, מצוין המאפיין אותו מצפים מהמערכת ללמוד. המערכת מנסה לחזות את המאפיין הרצוי של דוגמית המידע ולפי מוצא פונקציית ההפסד שלה, עורכת שינויים בשכבות השונות.

למידע מקוטלג זה, המשמש לאימון, קוראים training set.

משנסתיים שלב אימון המערכת, על מנת לאמוד את אחוז הדיוק שלה, מזינים אליה מאגר נתונים מקוטלג נוסף. המערכת מבצעת את אותו תהליך חיזוי, אך אינה מעדכנת את השכבות בהתאם למוצא פונקציית ההפסד. בשלב זה מחושב אחוז הדיוק של המערכת. מאגר מידע זה מכונה validation set [12].

3.2.8. פרמטרים חשובים במערכות לומדות

- קצב הלמידה (learning rate) [13] – קצב הלמידה מגדיר את גודל הצעד אותו עושים בכל שלב כדי להתקרב למינימום. קצב הלמידה יכול להיות קבוע או להתעדכן לאורך תהליך האופטימיזציה
- מחזורים (epochs) [13] – מגדיר את מספר הפעמים שמריצים את תהליך הלמידה והולידציה על המערכת.

4. זיהוי חולשות על פי תמונה

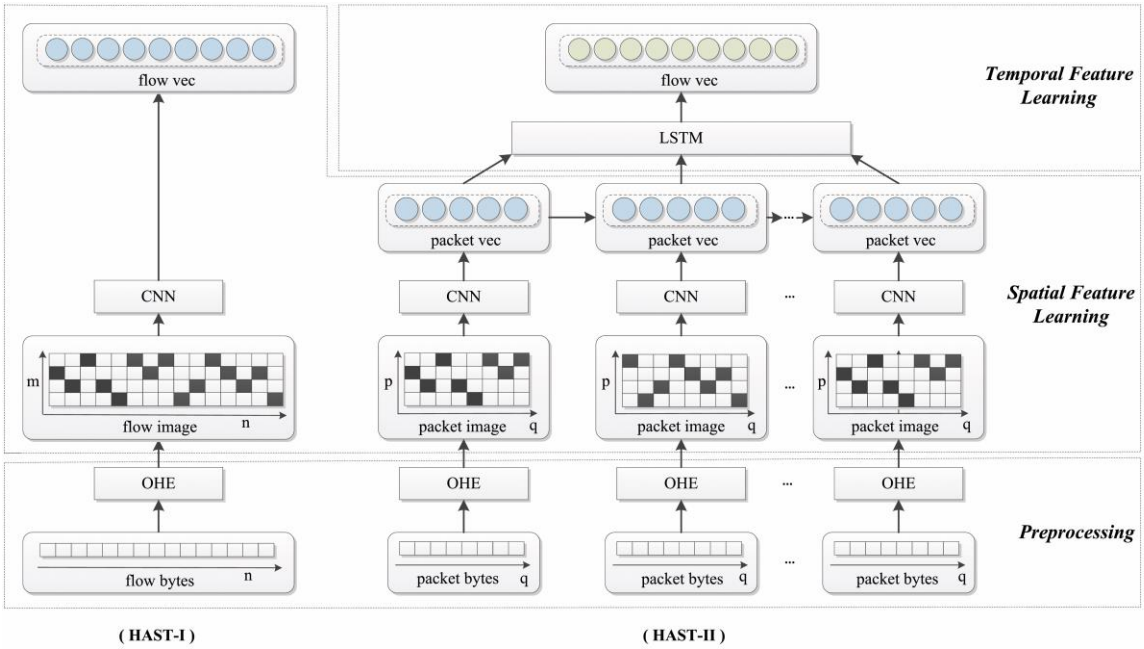
השילוב של מערכות לומדות במערכת זיהוי חדירות הינו נושא בו יש עיסוק רב בשנים האחרונות[15] [14] . בפרויקט זה ניסינו להתמודד עם שניים מהקשיים הקיימים ביצירת מערכת IDS בכלל ובשילוב הנ"ל בפרט: זיהוי חולשה על אף שהמידע עובר מוצפן וזיהוי השמשות שונות של אותה חולשה.

נשים לב שללא מעט מהחתימות המודרניות קיימים שני נופחים:

- זיהוי המבוסס על מידע קונקרטי המוכל בחולשה (חתימה סטאטית)
- זיהוי המבוסס על ההתנהגות הזמנית של החולשה (חתימה דינאמית - התנהגותית)

משמעות הדבר היא שייתכן שישנו מבנה קבוע הן מבחינה זמנית והן מבחינת צורנית (גודל ומבנה החבילות) עבור חולשה כלשהי, זאת מבלי תלות בהשמשה. על כן, במידה וניתן יהיה להמיר את ממד הזמן וממד הצורה לפורמט שנוח למערכת לומדת להתמודד איתה, דוגמת תמונה, ניתן יהיה לנסות לאמן מערכת לומדת לזהות השמשות שונות של חולשה כלשהי, גם כאשר זאת מוצפנת.

מחקר בנושא [16] יצר מימוש של מערכת שכזאת, אשר ממירה את המידע שעובר ברשת באופן הבא:



איור 6 - תיאור ויזואלי של תהליך עיבוד המידע במחקר

כפי שניתן לראות, במחקר זה, הוכנס זרמי מידע (data stream) אל המערכת, זאת לאחר שהמידע עובד לפורמט של תמונה, כאשר כל תמונה תויגה מראש לפי אופייה (בין אם מדובר בתקיפה מסוג מסוים ובין אם מדובר בחבילה תקינה). כיוון שאחוזי ההצלחה שהוצגו במחקר הנ"ל היו גבוהים והגיעו אף ל -99% ומעלה עבור חלק מהחולשות והתקיפות, בחרנו להשתמש במודל הנ"ל בכלל וספציפית במודל HAST-I כבסיס לפרויקט שלנו. מבנה המערכת HAST-I כפי שתואר במאמר הינו:

Layer	Type	Filters/neurons	Stride	Pad
1	conv+ReLU	32	1	same
2	max pooling	3	3	same
3	conv+ReLU	64	1	same
4	max pooling	3	3	same
5	dense	1,024	--	none
6	dense	5	--	none
7	softmax	--	--	none

איור 7 - השכבות השונות המרכיבות את HAST I

כמו כן, בחרנו להשתמש במסגרת הפרויקט בחולשת Bluekeep, אשר עוברת בצורה מוצפנת ואשר לה יש השמשות שונות (ואף חולשות דומות, אשר עשויות לשמש למחקרים עתידיים, עוד על כך בפרק הצעות להמשך עבודה). קישור לקוד המערכת מצוי בנספחים.

5. יצירת ה Data sets

על מנת ללמד את המערכת להבדיל בין תקשורת תקינה ותקשורת עוינת, אנחנו צריכים מאגר של תקשורת רלוונטית משני הסוגים. נכון לזמן כתיבת שורות אלו, כלל המאגרים הזמינים באופן פומבי, אשר היו עשויים להכיל את התקשורת הרלוונטית (בזכות האופן בו נאספו) נאספו טרם תחילת השימוש בפרוטוקול בו אנו עוסקים – RDP, או בסמוך לפרסומו, כאשר השימוש בו היה נדיר (ואכן, לא נמצא במאגרים הללו מידע רלוונטי).

עקב כך, עלה הצורך לבנות מאגר מידע משלנו בעזרתו נוכל ללמד את המערכת הלומדת.

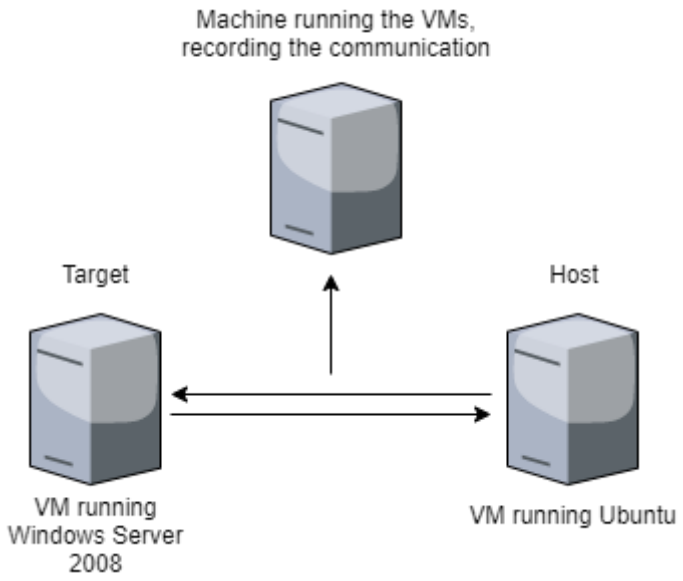
5.1 סביבת תקיפה

על מנת שנוכל ליצור תקשורת תקיפה היינו צריכים שתי סביבות - תקיפה ומטרה.

בעזרת VM Workstation Pro הרצנו שתי מערכות הפעלה במקביל:

- **Ubuntu** - שבה התקנו את כל הכלים שנצטרך לתקיפה - תוכנת הסנפה (tcpdump), פייתון, מטספלויט וכו…
- **Windows Server 2008** - מערכת המהווה אחת מהמערכות הפעלה שפגיעות לתקיפה שבחרנו להשתמש בה.

בעזרת מערכת ההפעלה אובונטו הרצנו את התקיפות שלנו (בעזרת מטספלויט או פייתון) אל המטרה, בזמן שאנחנו מקליטים את התקשורת בין השניים.

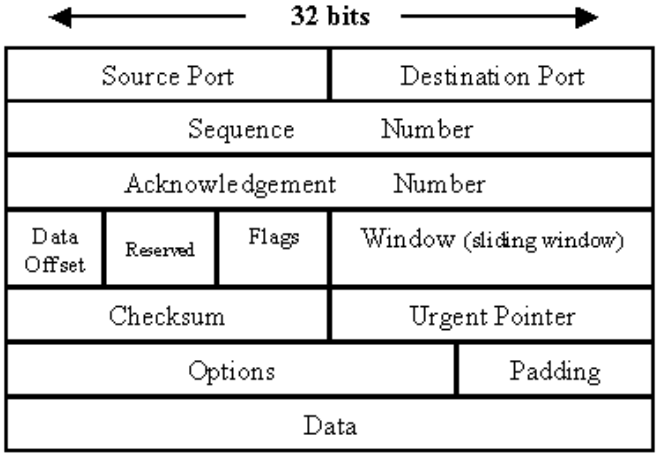


איור 8 - המחשה של סביבת התקיפה

5.2 תהליך חילוץ המידע (**parsing**)

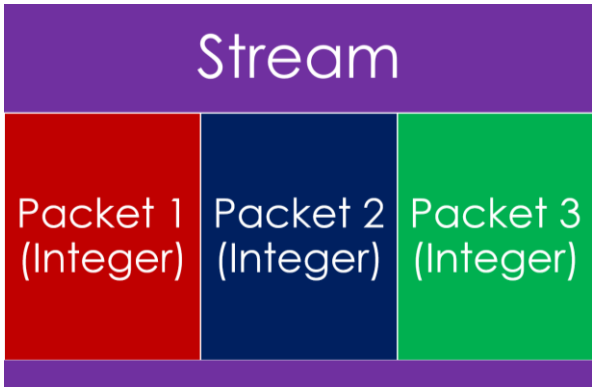
את תהליך חילוץ המידע מההסנפות ביצענו בדומה למתואר באיור [6]. כל צמד ערכי Hex תורגמו לערך דצימלי בתחום $[0\ 255]$, המקביל לתחום העוצמות המתאים עבור תמונת Grayscale.

את ערכי הפיקסלים שהתקבלו מילאנו במטריצות בגודל 48X32, כאשר תאים אותם לא ניתן היה למלא מפאת מחסור במידע, מילאנו ב-0. גודל המטריצות התקבל מהעובדה שגודל החבילות בסביבת התקיפה הוגבל ל [Byte]1514, כאשר $Ceil\left(\frac{1514}{32}\right) = 48$. נציין כי הבחירה ב 32 עמודות, נובע מהמבנה של TCP header:



איור 9 - המבנה של TCP Header

את המטריצות שהתקבלו שרשרנו ע"י הצבתן זאת לצד זאת לאורך ציר העמודות, כמתואר באיור הבא:



איור 10 - אופן שרשור החבילות ליצירת Stream

5.3 סוגי ה **Data-sets** השונים

על מנת לבדוק את המערכת היינו צריכים תקשורת תקיפה שממנה נלמד, תקשורת תקיפה אשר נבדוק אם אנחנו מצליחים לתייג, ותקשורת תקינה על מנת שהמערכת תוכל לסווג.

למטרה זו אספנו תקשורת מ מקורות שונים -

- תקשורת מתקיפה ע"י תוכנת Metasploit.
- תקשורת מתקיפה נוספת (נקרא לה 0xeb על שם היוצר), שאותה ננסה לגלות [17].
- תקשורת תקינה שהוקלטה מחיבור RDP ע"י תוכנת Remmina.
- תקשורת תקינה נוספת, הפעם מחיבור RDP ע"י תוכנת RDesktop.

מכל אחד מהסוגים, אספנו 1000 רצפים של תקשורת עבור לימוד המערכת, ועוד 100 עבור בדיקת יעילות המערכת לאחר האימון.

על מנת לראות איך המערכת תלמד הכי טוב, לימדנו את המערכת עם 2 קומבינציות של המקורות הנ"ל:

יחס סוגי החיבורים [%]	**הרכב הרצפים**
50/50	Metasploit/Remmina
50/25/25	Metasploit/Remmina/RDesktop

- נשים לב שהיחס הינו בהתאמה להרכב הרצפים
- השתמשנו בסה"כ 2000 חבילות ללימוד כל פעם.

הבדיקות בוצעו על 4 קומבינציות שונות של המקורות הנ"ל –

יחס סוגי החיבורים [%]	הרכב הרצפים
50/50	Metasploit/Remmina
50/25/25	Metasploit/Remmina/RDesktop
50/50	0xeb/Remmina
50/25/25	0xeb/Remmina/RDesktop

- כאשר השתמשנו בסה"כ 200 חבילות לבדיקה כל פעם.

6. אמצעים למניעת Overfitting

בתחילת העבודה על הפרויקט, התוצאות שהתקבלו הצביעו על אחוזי זיהוי גבוהים של המערכת הלומדת (מעל ל-95% של הפרדה בין חיבורים בטוחים לאלו בהם נעשה שימוש בחולשה). דבר זה העלה חשד בנוגע לאפשרות שהמערכת מזהה פרמטר כלשהו שאינו מוצפן, אליו לא התייחסנו, כאשר זיהוי זה יוצר overfitting – זיהוי של פרמטר ספציפי הייחודי ל-dataset ול -validation set בו אנו משתמשים, אך אינו בהכרח מאפיין את המידע אותו נרצה לזהות בעתיד. לאחר אנליזה מעמיקה של המידע שברשותנו, הצלחנו למצוא פרמטר שכזה.

כחלק מתהליך פתיחת קשר מוצפן המבוסס על פרוטוקולי SSL/TLS, שולח הלקוח מידע רב על אודות החיבור בצורה שאינה מוצפנת, אשר אותו ניתן להפוך, תוך שימוש באלגוריתם, למעין "טביעת אצבע" של החיבור. "טביעת אצבע" זאת קרויה JA3 SSL fingerprint, כאשר האלגוריתם בו משתמשים ליצירת "טביעת אצבע" זאת, לרבות קישור לקוד בו משתמשים לייצור לחילוצה, מפורט בבלוג [17].

תוך שימוש באלגוריתם הנ"ל, חילצנו את "טביעת האצבע" מהמידע שברשותנו:

JA3 SSL Fingerprint

User-Agent seen with the hash

004556e859f3c26c5d19746b3a957c74

-

- Metasploit 5.0.46-dev (BlueKeep) (count: 1, last seen: 2019-09-12 17:04:37)

ja3er.com query result

JA3 query result in ja3er.com website shows that hash belong to Metasploit Bluekeep exploit model. Below JA3 hashes obtained from my lab:

“004556e859f3c26c5d19746b3a957c74” – Metasploit Bluekeep exploit model

“53652b2730564404986852cde177b6d9” – Bluekeep rdpscan

איור11 - "טביעת האצבע" שהתקבלה מהמידע בו השתמשנו

כפי שניתן לראות, במידע הגלוי שברשותנו, מצויים פרטים גלויים המאפשרים את זיהוי החולשה. על מנת להימנע מ overfitting, החלטנו למחוק את כל המידע הגלוי מה-dataset ומה- validation set. את המחיקה ביצענו ע"י איפוס כלל המידע שעובר טרם סיום תהליך ה handshake של פרוטוקול ה-TLS.

בנוסף, על מנת לוודא שבמידע הגלוי שנותר, אין פרמטר המבדיל בין מידע השייך לחיבור בטוח לכזה ששייך לחיבור הכולל חולשה, בדקנו שאכן גרסאות ה-TLS בהן נעשה שימוש אינן מחולקות באופן דיכוטומי. במהלך הבדיקה זיהינו שימוש בשתי גרסאות TLS:

```
TLSv1.2 Record Layer: Change Cipher Spec Protocol: Change Cipher Spec
    Content Type: Change Cipher Spec (20)
    Version: TLS 1.2 (0x0303)
    Length: 1
    Change Cipher Spec Message

0000  00 50 56 ff 93 11 00 0c  29 27 87 a7 08 00 45 00   ·PV····· )'····E·
0010  00 a6 42 f2 40 00 40 06  ba 23 c0 a8 75 82 0d e1   ··B·@·@· ·#··u···
0020  f9 30 d4 5e 01 bb 24 1b  3c a6 70 97 23 ef 50 18   ·0·^··$· <·p·#·P·
0030  f5 3c c5 c8 00 00 16 03  03 00 46 10 00 00 42 41   ·<------ ··F···BA
0040  04 33 5a 0c 1e a0 33 51  15 f1 50 fe 36 0e f4 a6   ·3Z···3Q ··P·6···
0050  96 99 ff 88 8e 87 8d 10  f9 3e d8 c3 8a 4a 51 b6   -------- ·>···JQ·
0060  bb db 81 99 30 92 9b 01  2b 28 c9 4b 04 37 52 16   ····0··· +(·K·7R·
0070  3c df b7 fb 64 72 4b be  c3 5d 53 ee f5 1c ec ba   <···drK· ·]S-----
0080  8b 14 03 03 00 01 01 16  03 03 00 28 00 00 00 00   ------·· ···(····
```

```
TLSv1 Record Layer: Change Cipher Spec Protocol: Change Cipher Spec
    Content Type: Change Cipher Spec (20)
    Version: TLS 1.0 (0x0301)
    Length: 1
    Change Cipher Spec Message

0000  00 00 00 01 00 06 00 0c  29 c0 cb c0 00 00 08 00   -------- )-------
0010  45 00 00 6f 6e b8 40 00  80 06 08 37 c0 a8 01 27   E··on·@· ···7···'
0020  c0 a8 01 22 0d 3d 97 bb  83 17 e0 1a 1d d7 02 75   ···"·=·· -------u
0030  80 18 01 00 23 cf 00 00  01 01 08 0a 00 01 cf d7   ····#··· --------
0040  d1 5d cf 28 14 03 01 00  01 01 16 03 01 00 30 7a   ·]·(---- ------0z
```

איור12 - החלק בחבילה המעיד על גרסת ה-TLS בה נעשה שימוש

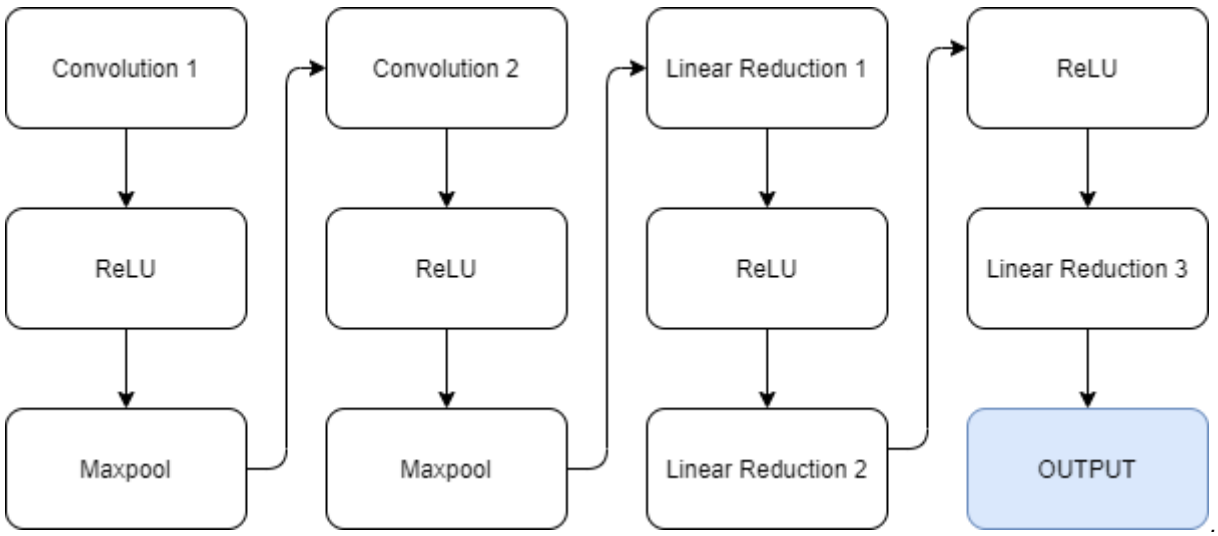
יחד עם זאת, מבדיקה מעמיקה, הסתבר כי ישנו שימוש בשתי גרסאות הפרוטוקול, הן בחיבורים בטוחים והן בחיבורים בהן נעשה שימוש בחולשה. מכאן, המשכנו בשימוש במידע המוצפן בלבד לאימון ובדיקת המערכת.

7. אימון הרשת ותוצאות

7.1. מבנה הרשת

המערכת בנויה ממספר שכבות משורשרות בטור אחת אחרי השנייה, כנזכר באיור [7] וכפי שמוצג באיור [13]. השכבות הנכללות הן:

- 2 שכבות קונבולוציה
- 4 שכבות ReLU
- 2 שכבות Maxpool
- 3 שכבות צמצום ליניארי



איור 13 - שרשור שכבות על מנת ליצור את מערכת הNN.

כאשר הפלט הסופי שלנו הוא וקטור מהצורה $\overrightarrow{Output} = \left(P(safe), P(exploit)\right)$.

הממדים של השכבות השונות הן:

Layer	*Dimensions*
Convolution 1	$1 \rightarrow 128\,,32$[1]
Convolution 2	$128 \rightarrow 32,4$
Maxpool	*(2, 2)*
Linear 1	$*^{2} \rightarrow 128$
Linear 2	$128 \rightarrow 32$
Linear 3	$32 \rightarrow 2$

7.2. <u>תצורות הרצה</u>

בנוסף לתצורות המידע השונות שתוארו ב- [5.3], הרצנו את הרשת תוך שימוש בפרמטרים שונים על מנת לבדוק מה מניב את התוצאות הכי טובות:

- שתי פונקציות אופטימיזציה שונות:
  - RMS prop – הפונקציה אשר בה השתמשו במחקר המקורי.
  - SGD – פונקציית אופטימיזציה בשימוש נפוץ במערכות NN.
- על מנת לוודא שהמערכת אינה מתמקדת בפיצ'ר אשר נמצא במקום קבוע בתקשורת, ביצענו הזזה רנדומלית שמאלה של התמונה[3]. שיטה זאת מהווה את אחת השיטות בעזרתן ניסינו להימנע מ – overfitting, כמוסבר לעיל. טווחי ההזזה בהם השתמשנו הינם:
  - 0
  - 1
  - 3
  - 5

יש לציין שעבור ערכי הזזה הגדולים מ-5 חבילות, קיבלנו תוצאות המצביעות, בקירוב, כי המערכת מנחשת (50% עבור כל אחת מהאפשרויות, ללא תלות בכניסות). כלומר, יש חסם עליון על גודל ההזזה שניתן לבצע למערכת ושהיא עדיין תצליח ללמוד ברמה נאותה.

---

[1] כמות ערוצי כניסה, כמות ערוצי יציאה, גודל הגרעין.

[2] גודל וקטור הכניסה משתנה בצורה משמעותית לפי ממדי המידע שאנחנו מזינים למערכת.

עבור שיטות למידה שונות (למשל, כמות החבילות שאנחנו דוחסים לכל תמונה) היינו צריכים להתאים ממדים שונים למערכת.

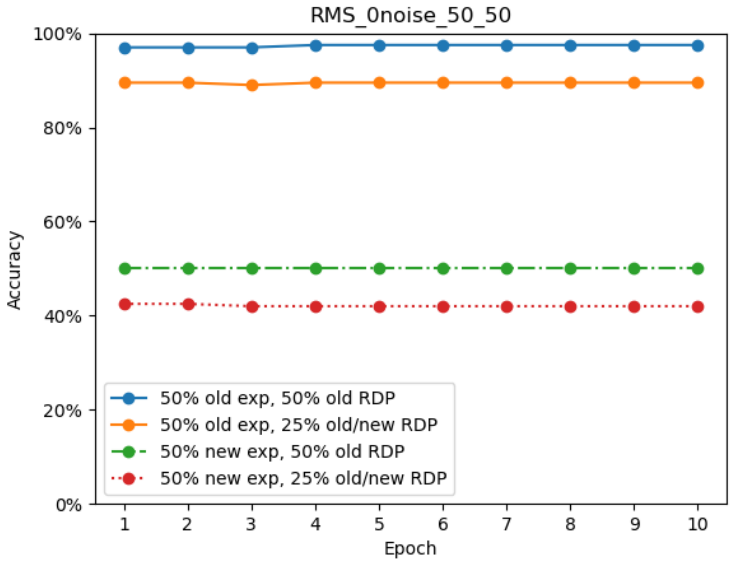
[3] הזזה שמאלה משמעה לקבוע מה תהיה החבילה הראשונה שנכנסת לתמונה. למשל, עבור הזזה של 5 חבילות, התמונה תתחיל מהחבילה השישית.

- גודל תמונה שונה – על אף שהסנפנו 200 חבילות בכל חיבור, ותוך הקפדה על כך שגם לאחר ההזזה המוזכרת לעיל, כמות החבילות אותן אנו מזינים לתמונה נשמרת, בדקנו מקרים בהם התמונה תכיל את כמויות החבילות הבאות:
  - 100
  - 128
  - 192
- לסיום, כמתואר בפרק 6, אימנו את המערכת על מידע מוצפן בלבד.

7.3. תוצאות

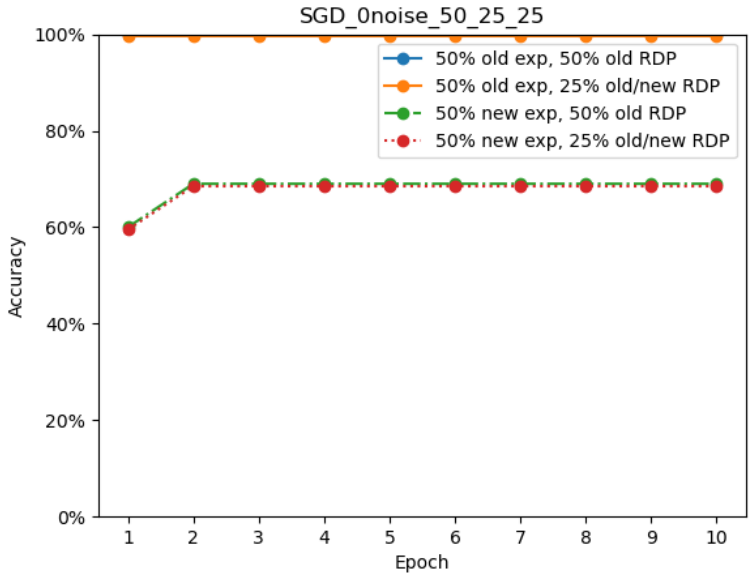
לאחר הרצת כלל הפרמוטציות השונות הגענו למסקנות הבאות:

- התוצאות הטובות ביותר התקבלו עבור 192 חבילות לכל תמונה.
- סוגים שונים של רעש אינם השפיעו באופן מורגש על התוצאות.
  - עובדה זאת חיזקה את האמונה שלנו שאין פיצ'ר ספציפי אשר הרשת מתמקדת בו.
- לא הייתה יתרון ביצועי לאחת מפונקציות האופטימיזציה.
- טרם הסרת המידע שאינו מוצפן, עבור מבנה נתונים שמכיל 50% תקיפת metasploit ו 50% תקשורת Remmina, הרשת נטתה לזהות אך ורק את תקיפת Metasploit כעוינת, בעוד שאת שאר התקשורת זיהתה כבטוחה. כלומר – המערכת ידעה לזהות היטב את המימוש של Metasploit, אך הניחה כי כל שאר התמונות מייצגות חיבורים בטוחים.



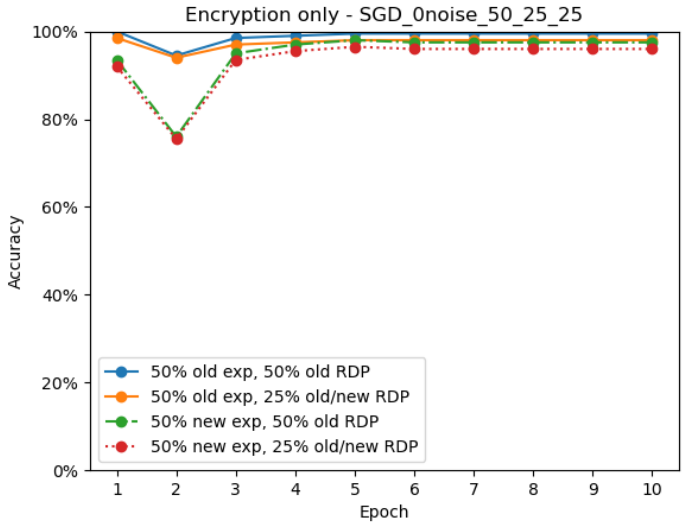
איור 14 –תמונות המורכבת מ 192 חבילות, ללא הזזה. המערכת מתקשה השמשת 0xeb

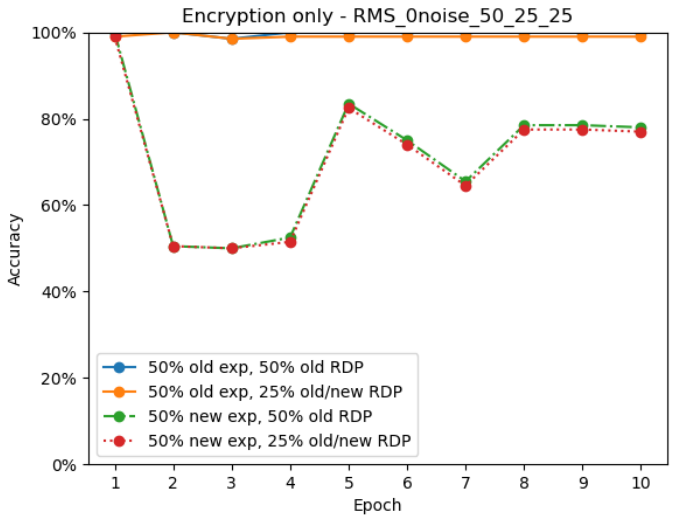
- לעומת זאת, עבור מבנה נתונים אשר מכיל 50% תקיפת Metasploit, 25% Remmina ו-25% RDesktop, המערכת זיהתה את תקשורת ה– Rdesktop כתקיפה. כלומר – המערכת ידעה לזהות היטב תקשורת בטוחה של Remmina, אך כל שאר התמונות זוהו כתקיפות.



איור 15 - תמונות המורכבת מ 192 חבילות, ללא הזזה. הרשת מתקשה בזיהוי חיבורי RDesktop

- לאחר הסרת המידע שאינו מוצפן, הגענו לתוצאות משמעותית יותר מוצלחות. עבור מבנה נתונים של 50\25\25 הנ"ל, קיבלנו דיוק של לפחות 75% לאורך הלמידה.





איור 16 - לאחר הסרת המידע שאינו מוצפן, המערכת מגיע לתוצאות המוכיחות יכולת זיהוי

8. מסקנות

8.1. היתכנות

מהתוצאות נראה שיש סיבה טובה להאמין שאפשרי ליצור חתימות רחבות אך ורק מהמידע המוצפן אותו ניתן להסניף מתקשורת ברשת. רוב התוצאות שקיבלנו היו משמעותית יותר טובות מ 50%, כלומר הן שיפור ניכר מניחוש טהור, מכאן שרשת נוירונים שבנינו אכן עובדת.

8.2. מקום לשיפור

אמנם התוצאות נראות מבטיחות, אך הן רחוקות מלהיות ברמה הדרושה על מנת להשתמש במערכת בעולם האמיתי. על מנת שנוכל לסמוך על המערכת במצב אמת, היא צריכה לאבחן נכון תקשורת בדיוק הרבה יותר גבוה (למשל, 95%+) .

אמנם אנחנו מתקרבים לאחוזי דיוק הנ"ל, אך יש מקום להמשיך ולנסות לשפר את הפרמטרים עד לקבלת תוצאות המתאימות לנדרש בשוק.

9. הצעות להמשך עבודה

9.1. המשך הפרויקט על DejaBlue

חולשות DejaBlue הינן חולשות נוספות בפרוטוקול RDP, אשר דומות במבנה שלהן לחולשת Bluekeep. כפי שציינו לעיל, כאשר מתגלה חולשה אחת, נעשים מאמצים רבים ע"י האקרים למציאת חולשות דומות לה. על כן, היכולת לזהות חולשות דומות, זאת בטרם שהן נחשפו בפומבי, יכולה להיות שוברת שוויון בעולם אבטחת המידע.

הקושי המרכזי בכיוון מחקר זה הוא לממש את חולשת DejaBlue. על אף שישנם תיאורים מפורטים של השמשות שונות, דוגמת [18], אין נכון לזמן כתיבת מילים אלו השמשה נגישה ושמישה ברשת.

9.2. הרחבת הפרויקט למספר סוגי חולשות

על מנת לייצר מערכת IDS יעילה, יש צורך בזיהוי מגוון רב של חולשות בעלות מאפיינים שונים. על מנת לוודא כי מערכת מהסוג אותו בדקנו אכן יכולה להוות בסיס למערכת IDS, יש לוודא את תוצאותיה על סוגים נוספים של חולשות, בעלי מאפיינים שונים.

הקושי המרכזי בכיוון מחקר זה הוא ביצירת Dataset מתאים, כפי שהצגנו לעיל.

9.3. מערכת לייצור Dataset

כפי שהזכרנו לעיל, קיימים נכון להיום קשיים רבים במציאת Dataset מתאים עבור מחקרים בתחום אבטחת המידע בכלל וספציפית עבור אבטחת מידע ברשתות מחשבים. על כן, מערכת אשר מייצרת סביבת תקיפה ומדמה את תהליך התקיפה, כך שניתן יהיה לאסוף נתונים למחקר, תהיינה שימושית.

הקושי המרכזי בייצור מערכת שכזאת הינה בשחזור תרחישי תקיפה אשר ידמו באופן אמין תרחישי תקיפה אמיתיים. עוד על הנושא ניתן לקרוא במאמר [19] .

9.4. מערכת לומדת לייצור חתימות

בהינתן שזוהתה תקיפה ברשת, ניתן לנסות ולייצר חתימה של המערכת הלומדת ע"י יצירת מערכת משוב בה המערכת הלומדת מצד אחד מנסה לייצר כל פעם וריאציות שונות של התקיפה, בעזרתן מנסים לתקוף מחשב יעד ומצד שני לומדת את ההסנפות המתקבלות מתהליכי התקיפה במטרה לנסות ולייצר חתימה רחבה של החולשה. קטלוג ההסנפות ייעשה על פי תוצאת התקיפה. בעזרת שיטה זאת, ניתן יהיה אף לגלות חולשות אשר דומות לחולשות מוכרות, אך טרם התגלו.

הקושי המרכזי בייצור מערכת שכזאת יהיה בבניית חלק המערכת שאחראי על ייצור וריאציות שונות על התקיפה. חלק זה ידרוש ידע רב בתחום אבטחת המידע על מנת לייעל את תהליך ייצור הוריאציות.

דוגמה למחקר דומה ניתן למצוא במאמר [20].

9.5. מימוש גרסה חומרתית של המערכת

ניתן לנסות ולממש את המערכת הסופית אותה קיבלנו בפרויקט על רכיב חומרתי תכנית (דוגמת FPGA) וכך לייצר מעין כרטיס רשת, אשר מאפשר העברה אך ורק של חבילות שאינן כוללות את החולשה הנ"ל.

הקושי המרכזי במימוש שכזה יהיה לאפשר לכרטיס הנ"ל לעמוד בקצבי הרשת, על אף החישובים הנדרשים עבור המערכת.

9.6. מערכת לגילוי חולשות Zero-Day

במידה וניתן יהיה ללמד מערכת כלשהי את כלל ההודעות ה"תקינות" האפשריות בפרוטוקול כלשהו, ניתן יהיה לנסות ולבדוק האם המערכת מצליחה לאתר חריגות כלשהן מהפרוטוקול, דבר אשר מצביע על שימוש פוטנציאלי בחולשה. מערכת שכזאת עשויה לתרום בזיהוי חולשות Zero-day, חולשות שאינן מוכרות באופן פומבי [21].

הקושי המרכזי במימוש מערכת יהיה בללמדה את כלל מנעד האפשרויות של פרוטוקול.

10. רשימות איורים

11. ביבליוגרפיה

[1] "Understanding the Remote Desktop Protocol (RDP)," 09 august 2020. [מקוון]. Available: https://docs.microsoft.com/en-us/troubleshoot/windows-server/remote/understanding-remote-desktop-protocol.

[2] E. Rescorla, "rfc5246 - The Transport Layer Security (TLS) Protocol," Network Working Group, 2008.

[3] "What Happens in a TLS Handshake? | SSL Handshake," CloudFlare, [מקוון]. Available: https://www.cloudflare.com/learning/ssl/what-happens-in-a-tls-handshake/.

[4] "Vulnerabilities and Exploits," enisa - European Union Agency for Cybersecurity, [מקוון]. Available: https://www.enisa.europa.eu/topics/csirts-in-europe/glossary/vulnerabilities-and-exploits.

[5] "What is an Intrusion Detection System (IDS)?," Checkpoint, [מקוון]. Available: https://www.checkpoint.com/cyber-hub/network-security/what-is-an-intrusion-detection-system-ids/.

[6] A. Malanov, "Antivirus fundamentals: Viruses, signatures, disinfection," Kaspersky, 13 October 2016. [מקוון]. Available: https://www.kaspersky.com/blog/signature-virus-disinfection/13233/.

[7] Cameron H. Malin, Eoghan Casey and James M. Aquilina, Malware Forensics Field Guide for Linux Systems, Syngress, 2014.

[8] Sean Dillon, Ryan Hanson, OJ Reeves, Brent Cook, "CVE-2019-0708 BlueKeep RDP Remote Windows Kernel Use After Free," Rapid 7, 23 September 2019. [מקוון]. Available: https://www.rapid7.com/db/modules/exploit/windows/rdp/cve_2019_0708_bluekeep_rce/.

[9] "Remote Desktop Services Remote Code Execution Vulnerability - CVE-2019-0708," Microsoft, 14 May 2019. [מקוון]. Available: https://msrc.microsoft.com/update-guide/vulnerability/CVE-2019-0708.

[10] "Metasploit Framework," Rapid7, [מקוון]. Available: https://docs.rapid7.com/metasploit/msf-overview/.

[11] Shai Ben-David, Understanding Machine Learning - From Theory to Algorithms, Cambridge: Cambridge University Press, 2014 & Shai Shalev-Shwartz.

[12] נדב בהונקר, שונית חביב ואורי בריט, "מבוא ללמידה עמוקה - ספר מעבדה," הטכניון, חיפה, 2020.

[13] S. Ruder, "An overview of gradient descent optimization," *arXiv*, 15 July 2017.

[14] Sherali Zeadally, Erwin Adi, Zubair Baig and Imran A. Khan, "Harnessing Artificial Intelligence Capabilities to Improve Cybersecurity," *IEEE Access*, כרך 8, 2020, pp. 23817 - 23837.

[15] Anna L. Buczak and Erhan Guven, "A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection," *IEEE Communications Surveys & Tutorials*, כרך 18, pp. 1153 - 1176, 2016.

[16] W. Wang, Y. Sheng, J. Wang, X. Zeng, X. Ye, Y. Huang and M. Zhu, "HAST-IDS: Learning Hierarchical Spatial-Temporal Features Using Deep Neural Networks to Improve Intrusion Detection," *IEEE Access*, כרך 6, 2018, pp. 1792-1806.

[17] J. Althouse, "TLS Fingerprinting with JA3 and JA3S," Salesforce Engineering, 15 January 2019. [מקוון]. Available: https://engineering.salesforce.com/tls-fingerprinting-with-ja3-and-ja3s-247362855967.

[18] "DejaBlue: Analyzing a RDP Heap Overflow," MalwareTech, 19 August 2019. [מקוון]. Available: https://www.malwaretech.com/2019/08/dejablue-analyzing-a-rdp-heap-overflow.html.

[19] Ali Shiravi, Hadi Shiravi, Mahbod, Tavallaee, Ali A.Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection", *Security & Computers*, כרך 31, מס' 3, 2012, pp. 357-374.

[20] Hu Zhengbing; Li Zhitang; Wu Junqi, "A Novel Network Intrusion Detection System (NIDS) Based on Signatures Search of Data Mining", ב- *First International Workshop on Knowledge Discovery and Data Mining*, Adelaide, SA, Australia, 2008.

[21] "Check Point", What is Zero Day Attack?, [מקוון]. Available: https://www.checkpoint.com/cyber-hub/threat-prevention/what-is-zero-day-attack/.

12. נספחים

- קישור ל-GIT של הפרויקט:

https://github.com/hallershahaf/AI_IDS