

# NetGuard



**מגישים:**

רועי דוד – 322526732

דניאל פרנקלך - 322403155

1.0	מבוא	3
1.1	תקציר של הפרויקט	3
1.2	מטרות ויעדים של הפרויקט	4
2.0	דרישות קדם	5
2.1	דרישות חומרה	5
2.2	דרישות תוכנה	5
2.3	הרשאות וגישות	5
3.0	הוראות התקנה	6
4.0	הוראות שימוש	7
4.1	הפעלת המערכת	7
4.2	כניסה לממשק המשתמש	7
4.3	ניהול המערכת מממשק המשתמש	8
4.3.1	ניהול חוקים	11
4.3.2	הדאשבורד	13
4.3.3	ניתוח התנהגות חריגה	15
5.0	ארכיטקטורת המערכת	17
5.1	כללי	17
5.2	מבנה המחלקות	18
5.3	מערכות היחסים בין המחלקות	21
5.4	מחלקות מרכזיות	23
6.0	תהליכים עיקריים	26
6.1	התהליך הראשי	26
6.1.1	הפעלת ממשק המשתמש מה Thread הראשי	27
6.1.2	הפעלת ה-Frontened	28
7.0	בדיקות	32
8.0	רעיונות לשיפור והרחבת המערכת	35
9.0	סיכום פרויקט NetGuard	37

## 1.0 מבוא

בעידן הדיגיטלי המודרני, מערכות מחשוב ורשתות תקשורת מהוות בסיס לפעילותם של ארגונים רבים. תעבורת הנתונים היומיומית המתרחשת ברשתות אלה מכילה מידע קריטי

ורגיש כמו פרטים אישיים של לקוחות, עסקאות פיננסיות, התכתבויות פנימיות ועוד. עם העלייה בכמות המידע אשר עובר ברשת, עולה גם הסיכון להתקפות סייבר, חדירות בלתי מורשות לרשת והשבתה של מערכות קריטיות על ידי גורמים עוינים.

אבטחת המידע ברשת הוא תחום המתמקד בהגנה על תשתיות תקשורת ומידע מפני איומים אלה. כיום, נדרשים כלים מתקדמים לניטור תיבורת הרשת, זיהוי איומים בזמן אמת, והגנה על משאבים קריטיים. אך עם העלייה במורכבות ובמגוון ההתקפות, משימת אבטחת הרשת הופכת להיות משימה מאתגרת יותר ויותר.

פרויקט NetGuard מתמקד בפיתוח מערכת חכמה לניטור וניהול תעבורה רשתית במחשבים, ומאפשרת זיהוי מהיר של תעבורה בלתי מורשית, ניתוח נתונים בזמן אמת וניהול חוקים. המערכת נועה לאפשר לארגונים לשמור על מכשירי הקצה שלהם ועל אבטחת המידע, לזהות איומים אפשריים באופן מיידי ולהגיב אליהם במהירות.

## 1.1 תקציר של הפרויקט

פרויקט NetGuard מאפשר ניטור על התעבורה הנכנסת והיוצאת במחשב, וכוללת זיהוי וניהול של חוקים, ניתוח תעבורת רשת בזמן אמת, קבלת התראות בזמן אמת על פקטות חשודות, ניטור חריגות ברשת והצגת נתונים על ידי לוח בקרה גרפי.

המערכת מאפשרת למשתמשים להגדיר חוקים לניהול תעבורת הרשת, להציג סטטיסטיקות בזמן אמת על תעבורת נתונים ולהתריע בצורה מהירה על חריגה או איום בתעבורה הרשתית. בנוסף, המערכת מאפשרת להוסיף, לערוך ולמחוק חוקים בהתאם לצרכי המשתמש.

פיתוח המערכת נעשה תוך שימוש במגוון כלים וטכנולוגיות, וכתובה בשפת python. המערכת כוללת אכסון נתונים בMongoDB בעזרת הספרייה pymongo, מעקב אחר התעבורה הרשתית בעזרת ספריית scrapy, Uli אינטראקטיבי ונוח בעזרת הספרייה pywebio ושימוש בספריית dash להצגת נתונים וסטטיסטיקות בצורה גרפית נוחה.

## 1.2 מטרות ויעדים של הפרויקט

### מטרות הפרויקט

1. **ניהול ואבטחת תעבורת רשת** - לאפשר מעקב והגנה על המחשב האישי או על הרשת על ידי ניטור תעבורה נכנסת ויוצאת, זיהוי פקטות חריגות או מסוכנות, וניהול חוקים מותאמים אישית לאבטחה.
2. **זיהוי איומים בזמן אמת** - להתריע למשתמש או למנהל המערכת על פעילות חריגה או זדונית בזמן אמת על מנת לאפשר תגובה מיידית ומניעת נזק.
3. **שיפור היעילות של ניטור רשת** - מתן ממשק משתמש יעיל וברור לניהול חוקים, סינון נתונים, וניטור דוחות סטטיסטיים על מנת לפשט את תהליך האבטחה והפיקוח על תעבורת הרשת.

### יעדי הפרויקט

1. **מעקב וניטור תעבורה** - לנתח תעבורה נכנסת ויוצאת, להציג סטטיסטיקות חיות, ולספק נתונים מפורטים על כל פקטה כולל מקור ויעד, פרוטוקול, ופורטים.
2. **ניהול חוקים מותאמים אישית** - להעניק אפשרות ליצירה, עריכה והסרה של חוקים אשר מגבילים או מתירים סוגי תעבורה מסוימים, ולהגדיר תנאים לחסימה או מתן התראה במקרים חריגים.
3. **יצירת דאשבורד להצגת סטטיסטיקות** - לפתח ממשק משתמש המציג מידע חזותי על סטטיסטיקות התעבורה, הפקטות, והאנומליות שנמצאו ברשת, כולל אפשרויות סינון נתונים למעקב ולשליטה.
4. **איתור אנומליות ואיומים** - לפתח מערכת חכמה לזיהוי חריגות כמו מספר גבוה של פקטות מ-IP מסוים או תעבורה רבה מדי בפרוטוקול או פורט מסוים, ולהציג התראות על פעילות חשודה.
5. **תיעוד התראות על פי חוקים** - לספק לוג מפורט של כל הפעילויות שנעשו בעקבות הפרות חוקים או זיהוי אנומליות, כולל אישור והתעדכנות במאגר לאחר בחינה של המנהל.
6. **יצירת ממשק משתמש נוח ואינטראקטיבי** - לפתח ממשק משתמש ידידותי המאפשר אינטראקציה חלקה עם המערכת, כולל יכולות סינון, חיפוש והתאמה אישית, על מנת לשפר את חוויית המשתמש ולייעל את תהליך הניהול.

## 2.0 דרישות קדם

### 2.1 דרישות חומרה

- **מעבד:** מעבד עם תמיכה ב-multithreading לביצוע מספר Threads במקביל.
- **זיכרון RAM:** לפחות 4GB RAM להבטחת פעולה תקינה של המערכת בזמן ניתוח תעבורה.
- **כונן קשיח:** לפחות 10GB אחסון פנוי לצורך שמירת נתונים ותעבורת רשת ב-Database.
- **רשת:** חיבור אינטרנט יציב וגישה לרשת המקומית לניתוח תעבורת הרשת.

### 2.2 דרישות תוכנה

- **מערכת הפעלה:** MacOS או Linux/Unix
- **Python:** גרסה 3.8 ומעלה.
- **MongoDB:** בסיס נתונים NoSQL לאחסון נתוני התעבורה, הכללים והאנומליות.
- **ספריות Python:**
  - Scapy: לניטור תעבורת רשת.
  - pywebio: לפיתוח ממשק משתמש מבוסס Web.
  - pymongo: לניהול ושמירת נתונים ב-MongoDB.
  - dash ו-plotly: ליצירת גרפים ודשבורדים להצגת נתונים.
  - matplotlib - ליצירת גרפים ויזואליים
  - web3 - לתקשורת עם רשתות בלוקצ'יין כמו את'ריום
  - flask - ליצירת אפליקציות ווב קלות
- ספריות נוספות כמו datetime, threading, functools ו-logging (מגיעות עם Python).

### 2.3 הרשאות וגישות

- **הרשאות מערכת:** הפעלת המערכת דורשת הרשאות סופר-יוזר (sudo) על מנת לנטר את תעבורת הרשת.
- **גישה לרשת:** גישה מלאה לאינטרנט ולרשת המקומית בה פועל NetGuard לצורך ניטור וניתוח תעבורת הרשת.

## 3.0 הוראות התקנה

### 1. וודא שיש לך את Python3 מותקן:

תוכל לבדוק אם Python מותקן על ידי הפקודה:

```
python --version
```

### 2. צור סביבה וירטואלית (מומלץ, אך לא חובה):

○ סביבה וירטואלית תסייע לשמור את התלויות של הפרויקט מבודדות.

○ צור סביבה חדשה על ידי הפקודה:

```
python -m venv venv
```

i. הפעל את הסביבה (ב-Linux או macOS):

```
source venv/bin/activate
```

ii. Windows:

```
venv\Scripts\activate
```

### 3. התקן את התלויות:

○ גש לתיקיית requirements והתקן את כל הספריות הנדרשות על ידי הפקודה:

```
pip install -r requirements.txt
```

הקובץ requirements.txt מכיל את כל הספריות הנדרשות לטובת ריצת הפרויקט

## 4.0 הוראות שימוש

### 4.1 הפעלת המערכת

הרצת סקריפט להפעלת המערכת

יש להריץ את הסקריפט הבא:

`./start_netguard.sh`

```
(base) → NetGuard git:(danielle) ✕ ./start_netguard.sh
Password:
Log file cleared.
Rule ID    Source IP      Destination IP  Source Port    Destination Port Protocol  Action
=====
1          10.0.0.5       N/A            N/A            N/A            N/A      block
2          N/A            10.0.0.255     N/A            N/A            N/A      block
Running on all addresses.
Use http://192.168.10.30:8088/ to access the application
```

הסקריפט הזה מוודא שהמערכת תורץ עם ההרשאות הנדרשות, ומפעיל את כל השירותים הדרושים.

לאחר הרצת הסקריפט, צריך להזין את הסיסמה של המחשב עבור ריצת `sudo`.

### 4.2 כניסה לממשק המשתמש

לאחר הרצת הסקריפט, ניתן לגשת לממשק המשתמש של המערכת באמצעות דפדפן. יש לפתוח את הכתובת הבאה: `/http://localhost:8088`

כניסה לכתובת זו יאפשר לצפות בלוח הבקרה ולבצע פעולות ניהוליות במערכת כמו ניתוח פקטות, ניהול חוקים, וצפייה באנומליות - כלל הפעולות יפורטו בהמשך.

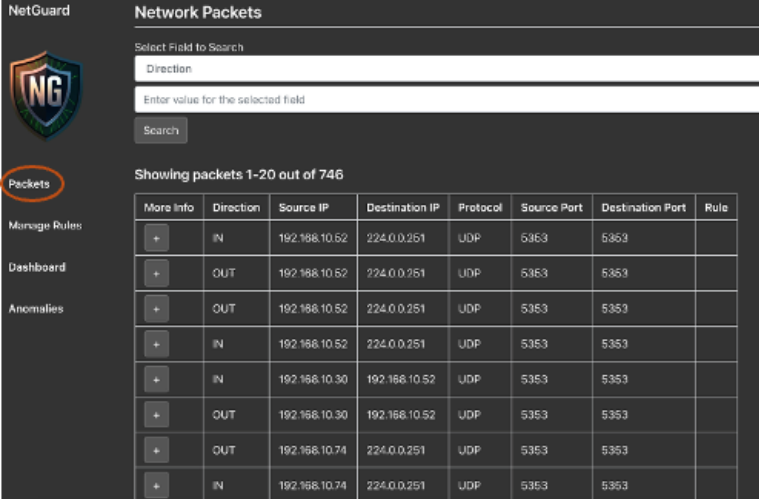
## 4.3 ניהול המערכת מממשק המשתמש

### צפייה בפקטות ופילטורים

היעד הראשון של המערכת NetGuard הוא מעקב וניטור אחר התעבורה הרשתית למחשב ומהמחשב, על מנת לאתר פעילויות חריגות, לזהות ניסיונות פריצה או התנהגות חשודה ולספק התראות על אירועים אלה בזמן אמת.

### צפייה בפקטות

במסך הראשי, כאשר נכנסים לאתר לראשונה, העמוד "Packets" נפתח אוטומטית. אם רוצים לחזור לעמוד זה, יש ללחוץ על הלשונית בשם "Packets" בתפריט הצדדי.

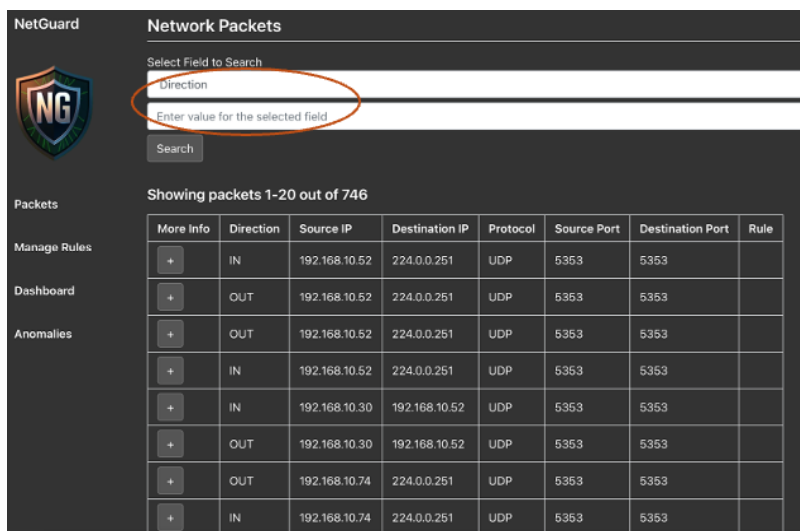


Network Packets							
Select Field to Search							
Direction							
Enter value for the selected field							
Search							
Showing packets 1-20 out of 746							
More Info	Direction	Source IP	Destination IP	Protocol	Source Port	Destination Port	Rule
+	IN	192.168.10.62	224.0.0.251	UDP	5353	5353	
+	OUT	192.168.10.62	224.0.0.251	UDP	5353	5353	
+	OUT	192.168.10.62	224.0.0.251	UDP	5353	5353	
+	IN	192.168.10.62	224.0.0.251	UDP	5353	5353	
+	IN	192.168.10.30	192.168.10.52	UDP	5353	5353	
+	OUT	192.168.10.30	192.168.10.52	UDP	5353	5353	
+	OUT	192.168.10.74	224.0.0.251	UDP	5353	5353	
+	IN	192.168.10.74	224.0.0.251	UDP	5353	5353	



## חיפוש וסינון

1. בחלק העליון של דף ה-Packets, יש אפשרות לבחור שדה לחיפוש מסוים.
2. **Select Field to Search** - בחר את השדה שלפיו תרצה לסנן את הפקטות (למשל: כיוון הפקטה - OUT/IN, כתובת IP מקורית או יעד, פרוטוקול, חוק שהפקטה התאמתה עליו ועוד).
3. **Enter value for the selected field** - הקלד את הערך שלפיו תרצה לסנן את תוצאות הפקטות.
4. לחיצה על **Search** תציג את התוצאות המתאימות בחלון התוצאות מתחת.



NetGuard Network Packets

Select Field to Search: Direction

Enter value for the selected field

Search

Showing packets 1-20 out of 746

More Info	Direction	Source IP	Destination IP	Protocol	Source Port	Destination Port	Rule
+	IN	192.168.10.52	224.0.0.251	UDP	5353	5353	
+	OUT	192.168.10.52	224.0.0.251	UDP	5353	5353	
+	OUT	192.168.10.52	224.0.0.251	UDP	5353	5353	
+	IN	192.168.10.52	224.0.0.251	UDP	5353	5353	
+	IN	192.168.10.30	192.168.10.52	UDP	5353	5353	
+	OUT	192.168.10.30	192.168.10.52	UDP	5353	5353	
+	OUT	192.168.10.74	224.0.0.251	UDP	5353	5353	
+	IN	192.168.10.74	224.0.0.251	UDP	5353	5353	

## תצוגת רשימת פקטות

לאחר החיפוש או טעינת התצוגה הראשונית, יופיעו הפקטות עם המידע הבא:

- **More Info**: כפתור "+" המאפשר להציג מידע מפורט נוסף על הפקטה (Packet Info) שלא נמצא בדף הראשי כגון payload, גודל הפקטה, ttl ועוד.
- **Direction**: הכיוון של הפקטה (IN עבור נכנסת, OUT עבור יוצאת).
- **Source IP**: כתובת ה-IP שממנה הגיעה הפקטה.
- **Destination IP**: כתובת ה-IP שאליה מיועדת הפקטה.
- **Protocol**: סוג הפרוטוקול (TCP, UDP וכו').
- **Source Port** ו-**Destination Port**: פורטים מקוריים ויעדיים של הפקטה.
- **Rule**: החוק שהתאים לפקטה, אם קיים.

הכפתור תחת "More Info" יביא מידע נוסף על הפקטה:

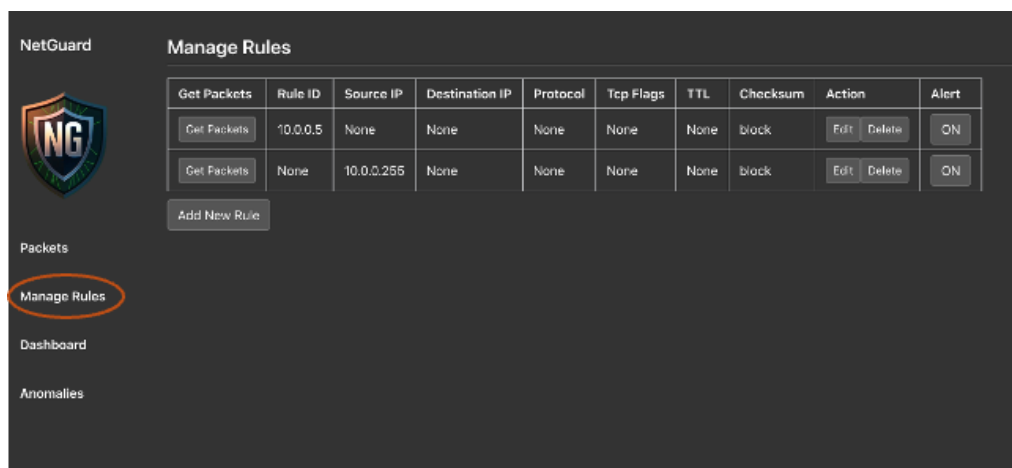
Packet Info	
Field	Value
_id	672a5f150451df33e334be29
Direction	OUT
Source IP	159.124.39.24
Destination IP	192.168.10.30
Protocol	TCP
Protocol Number	6
TTL	50
Packet Length	66
Source Port	443
Destination Port	57145
Matched Rule ID	
Frame Offset	0
More Fragments	
Payload	
Insertion Time	2024-11-05 20:08:21.921000

### 4.3.1 ניהול חוקים

היעד השני עבור הפרויקט הוא ניהול חוקים מותאמים אישית במטרה להגביל או להותיר תעבורה מסויימת, ולאפשר התראות עקב תעבורה שעונה על אחד החוקים

### צפייה בחוקים

בעת לחיצה על העמוד "Manage Rules" ניתן לראות את כלל החוקים אשר כבר קיימים במערכת. כל חוק מוצג בטבלה עם הפרטים שלו, כגון IP מקור, IP יעד, פרוטוקול, TTL (זמן חיים), פעולה (חסום/אפשר), ועוד.



### עריכת חוק קיים

בלחיצה על כפתור "Edit" ליד כל חוק, ניתן לערוך את פרטי החוק. יש לבצע את השינויים הדרושים ולשמור.

### מחיקת חוק קיים

בלחיצה על כפתור "Delete" ליד כל חוק, ניתן להסיר את החוק ממערכת ניהול התעבורה.

### הוספה של חוק חדש

לחיצה על הכפתור "Add New Rule" מאפשרת להוסיף חוק חדש. בחלון שייפתח, יש למלא את הפרטים הדרושים לחוק ולשמור את השינויים. אם נרצה שהחוק יהיה בעל פרמטרים נוספים, ניתן ללחוץ על הכפתור "Advanced" ושם מוצגים אפשרויות נוספים לשדות רלוונטיים עבור החוק.

בעת לחיצה על "Submit" החוק נשמר ב-MongoDB והמסך יחזור לרשימת החוקים הכוללת גם את החוק החדש.

## מעקב אחר פקטות התואמות את החוק

בלחיצה על הכפתור "Get Packets" המסך יעבור למסך "Packets" עם הסינון על החוק, וכך יוצגו הפקטות שתואמות את החוק הספציפי.

## התראות

בכל חוק ניתן לראות את המצב של התראות (Alert) המוגדרות לאותו החוק. במצב ON – התראות מופעלות, ובמצב OFF – התראות מושבתות. אם המשתמש מעוניין בקבלה או הפסקה של ההתראות, עליו ללחוץ על הכפתור עם המילה ON או OFF בהתאמה.

ההתראות יאפשרו למשתמש בזמן אמת לדעת האם נראתה פקטה אשר עונה על החוק הספציפי, ובכך לאפשר ערנות וחדות על המתרחש בתעבורה בקלות רבה.

Manage Rules									
Get Packets	Rule ID	Source IP	Destination IP	Protocol	Tcp Flags	TTL	Checksum	Action	Alert
Get Packets	10.0.0.5	None	None	None	None	None	block	Edit Delete	ON
Get Packets	None	10.0.0.255	None	None	None	None	block	Edit Delete	ON

Add New Rule

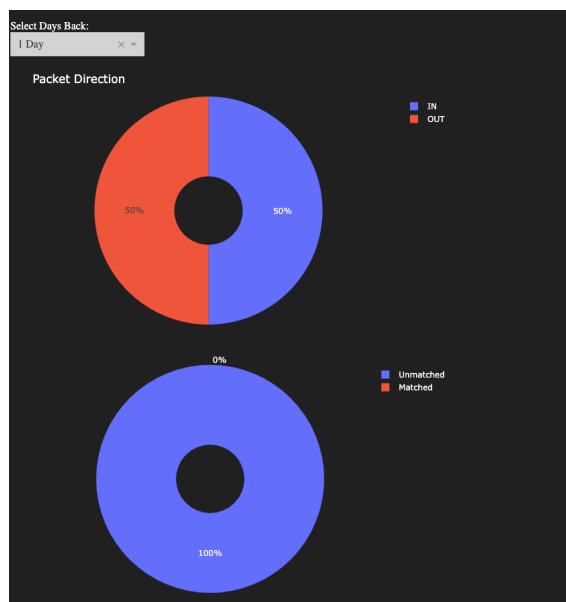
## 4.3.2 הדאשבורד

במסך הדאשבורד ניתן לנתח נתוני רשת באמצעות מגוון גרפים וכלים ויזואליים המספקים מבט כללי על תעבורת הרשת במערכת.

באמצעות הדאשבורד ניתן לקבל מבט על פעילות הרשת ולזהות שינויים, חריגות ודפוסים שיכולים להצביע על בעיות פוטנציאליות או איומים.

### צפייה בדאשבורד

בעת לחיצה על tab בשם "Dashboard" יתקבל מסך הדאשבורד. בו ניתן לראות את כלל הסטטיסטיקות המוצגות, ובהתאם לנדרש ניתן להוסיף בקוד דשבורדים נוספים במידת הצורך של לקוחות המערכת.



### בחירת פרק זמן לניתוח

בראש המסך מופיעה אפשרות לבחירת מספר הימים אחורה לצורך הצגת הנתונים, לדוגמה: יום אחד, שלושה ימים, או שבוע. שינוי הבחירה יעדכן את הנתונים בגרפים בהתאם לטווח הזמן שנבחר.

## **הגרפים הקיימים:**

### **1. חלוקת כיוון התעבורה**

הגרף הראשון מציג את כיוון התעבורה (Packet Direction) ומחלק אותה ל-"IN" (תעבורה נכנסת) ו-"OUT" (תעבורה יוצאת) לפי יחס אחוזים. זה מאפשר לקבל מושג כללי על חלוקת התעבורה ברשת לאורך התקופה שנבחרה.

### **2. כמות הפקטות לשעה**

גרף זה מציג את כמות הפקטות לאורך ציר הזמן, מחולק לשעות. בעזרת גרף זה ניתן לראות אם יש שינויים במידת התעבורה ברשת בשעות מסוימות לאורך היום.

### **3. פקטות לפי חוקים**

גרף נוסף מציג את חלוקת הפקטות על פי חוקים שהוגדרו במערכת. הוא מחלק את הפקטות ל-"Matched" (מתאימות לחוק מסוים) ו-"Unmatched" (שאינן תואמות חוקים). חלוקה זו מאפשרת לראות כמה מהתעבורה עומדת בקריטריונים שהוגדרו.

### **4. פקטות לפי התאמות חוקים**


בגרף זה ניתן לראות את החלוקה של הפקטות לפי התאמתן לחוקים הספציפיים שהוגדרו במערכת NetGuard. כל פקטה נבדקת לפי מספר חוקים מוגדרים מראש, שכל אחד מהם מתייחס לקריטריונים שונים. הגרף מאפשר למנהלי רשת להבין אילו חוקים מופעלים באופן תדיר ומתי, וכך לעקוב אחרי תבניות של תעבורה חשודה או תקינה בהתאם לחוקים שהוגדרו.

### 4.3.3 ניתוח התנהגות חריגה

המערכת יודעת להציג תעבורת רשת חריגה על ידי שימוש במידע המתקבל מאנומליות שזוהו. כל אנומליה שמתגלה ניתנת לצפייה בנפרד תחת עמוד האנומליות.

NetGuard

Anomalies



Packets

Manage Rules

Dashboard

Anomalies

Anomaly Name	Anomaly Time	Detail	Count
packets_from_same_ip_anomaly	2024-11-01 11:59:36	192.168.10.30	118
packets_from_same_ip_anomaly	2024-11-01 11:59:36	44.234.198.184	34
packets_from_same_ip_anomaly	2024-11-01 11:59:36	66.203.125.16	18
packets_from_same_ip_anomaly	2024-11-01 11:59:36	192.168.10.52	18
packets_from_same_ip_anomaly	2024-11-01 11:59:36	192.168.10.1	12
packets_from_same_ip_anomaly	2024-11-01 11:59:36	192.168.10.44	8
packets_from_same_ip_anomaly	2024-11-01 11:59:36	149.154.167.92	8
packets_from_same_ip_anomaly	2024-11-01 11:59:36	8.8.4.4	8
packets_from_same_ip_anomaly	2024-11-01 11:59:36	192.168.10.20	4
packets_from_same_ip_anomaly	2024-11-01 11:59:36	44.240.52.117	4
packets_from_same_ip_anomaly	2024-11-01 11:59:36	192.168.10.106	4
packets_from_same_ip_anomaly	2024-11-01 11:59:36	35.227.233.235	2
packets_from_same_ip_anomaly	2024-11-01 11:59:36	35.81.90.104	2
packets_from_same_ip_anomaly	2024-11-01 11:59:36	76.223.31.44	2
high_traffic_on_any_port	2024-11-01 11:59:36	443	116
high_traffic_on_any_port	2024-11-01 11:59:36	61461	34

### צפייה באנומליות

בעת לחיצה על ה-Tab בשם "Anomalies" ייפתח מסך שמאפשר צפייה בכלל האנומליות שהמערכת זיהתה, יחד עם מידע מפורט על כל אנומליה. מטרת מערכת NetGuard היא לספק תובנות על פעילות רשת לא רגילה שיכולה להעיד על איומים פוטנציאליים, ולכן היא יודעת לזהות את האנומליות הבאות:

1. **Packets\_from\_same\_ip\_anomaly** - אנומליה זו מתרחשת כאשר ישנה כמות רבה של פקטות המגיעות מאותה כתובת IP בפרק זמן קצר. מצב זה יכול להעיד על ניסיון לתקיפה מסוג Brute Force או DDoS, שבו נעשה ניסיון לשלוח כמויות גדולות של בקשות מאותו מקור כדי להציף את השרת או להתיש את המשאבים שלו.
2. **High\_traffic\_on\_any\_port** - אנומליה זו מזהה עומס בלתי רגיל בתעבורת הרשת על פורט מסוים. כשנרשמת תעבורה מרובה על פורט ספציפי, הדבר עשוי להצביע על ניסיון לנצל את הפורט לצורכי תקיפה, כגון חדירה דרך פורט פתוח או תקיפה מסוג Port Scanning, בה תוקף סורק פורטים כדי לאתר נקודות חולשה במערכת.
3. **high\_distinct\_destinations\_anomaly** - זוהי אנומליה שמתייחסת לכמות גדולה של פניות לכתובת IP, בטווח זמן קצר. מצב זה יכול להצביע על תופעה של Botnet או פעילות זדונית אחרת, שבה גורם עוין מפעיל רשת של מחשבים נגועים כדי לפנות לשרת מסוים, בדרך כלל לצורך התקפת DDoS או לצורך איסוף מידע.

הדיווח על אנומליות אלו מסייע למנהלי רשת לזהות פעילות חריגה ולטפל בה בזמן אמת, תוך הבנת דפוסי תעבורה שעשויים להעיד על איומים פוטנציאליים.

בכל אנומליה ניתן לצפות במידע מפורט הכולל את סוג האנומליה, זמן הזיהוי, פרטים רלוונטיים כגון כתובת ה-IP או הפורט המעורבים באנומליה, ואת כמות הפקטות שזוהו במסגרת אותה אנומליה. מידע זה מאפשר למנהלי הרשת להבין את מקור הפעילות החריגה ואת היקפה, ולהגיב בהתאם לאיומים פוטנציאליים.



## 5.0 ארכיטקטורת המערכת

### 5.1 כללי

#### 1. קובץ ההרצה הראשי (main.py)

- קובץ זה מהווה את נקודת הכניסה הראשית למערכת ומתחיל את כל תהליכי הרצת הפרויקט. הקובץ טוען את כל הרכיבים הנדרשים, כולל חיבור למסד נתונים, ניהול החוקים, והפעלת ממשק המשתמש.
- בעת ההפעלה, הקובץ פותח חיבור למסד הנתונים, טוען את כללי הסינון ומתחיל להאזין לתעבורת רשת באמצעות ממשקי רשת.

#### 2. ניהול החוקים (RuleSet)

- רכיב ניהול החוקים אחראי על שמירה, טעינה ובדיקת החוקים שעל פיהם מבוצע ניתוח התעבורה.
- כל חוק מכיל שדות כמו כתובת מקור, כתובת יעד, פרוטוקול ופעולה נדרשת (כגון חסימה). כאשר חבילת רשת מתקבלת, היא נבדקת מול סט החוקים כדי לקבוע אם היא תואמת לאחד מהם ומה הפעולה שיש לבצע בהתאם.

#### 3. חיבור למסד הנתונים (MongoDbClient)

- המערכת עושה שימוש ב-MongoDB לשמירת הנתונים שנאספו במהלך ניטור התעבורה.
- המידע הנשמר כולל חבילות חשודות, נתוני התראות והיסטוריה של הפעולות שננקטו. רכיב זה אחראי על החיבור למסד הנתונים וביצוע פעולות קריאה וכתובה בצורה מאובטחת.

#### 4. שכבת הניטור (network\_interface)

- רכיב זה אחראי על לכידת תעבורת הרשת, המבוצעת בעזרת ספריית Scapy.
- המערכת מגדירה ממשק רשת ספציפי אשר דרכו היא מנטרת את התעבורה הנכנסת והיוצאת. המערכת פועלת במצב אסינכרוני באמצעות מספר תהליכונים (Threads) במקביל, מה שמבטיח ניטור רציף ללא עיכובים.

#### 5. עיבוד חבילות ותיאום עם סט החוקים

- כל חבילת רשת שנלכדת מועברת לפונקציה manage\_sniffed\_packet לצורך עיבוד ובדיקת התאמה לחוקים.

- אם נמצאה התאמה לאחד מהחוקים המוגדרים, החבילה תתוייג כ"חשודה", ותישלח קריאה לפעולה (כגון חסימה, שליחת התראה או שמירה במסד הנתונים).

## 6. ממשק משתמש (ui\_module ו-dashboards.py)

- ui\_module ו-dashboards.py אחראים על הצגת המידע למשתמש דרך ממשק גרפי (בפריקט זה נראה כי נעשה שימוש ב-PyWebIO).
- הממשק מאפשר למשתמש לעקוב אחרי התראות, לעיין בנתונים שנשמרו במסד הנתונים ולראות מידע בזמן אמת על פעולות המערכת.

## 7. ממשק שורת הפקודה (סקריפט python-sudo.sh)

- סקריפט זה משמש להרצת התוכנית הראשית (main.py) עם הרשאות מוגבהות (sudo), דבר הנדרש כדי לאפשר גישה לממשק הרשת הנדרש ולפקודות מערכת מתקדמות.
- הסקריפט מפשט את ההרצה עבור המשתמש, כך שניתן להפעיל את המערכת בצורה קלה ועם ההרשאות הנכונות.

## 5.2 מבנה המחלקות

### 1. MongoClient MongoDB

- **תיאור:** מנהל חיבור למסד הנתונים MongoDB ומבצע פעולות CRUD (יצירה, קריאה, עדכון ומחיקה) עבור מסמכים שונים.

### • פונקציות עיקריות:

- insert\_to\_db: הוספת מסמך למסד.
- update\_in\_db: עדכון מסמך במסד.
- delete\_from\_db: מחיקת מסמך מהמסד.
- anomaly\_query: ביצוע שאילתות מותאמות לגילוי אנומליות.
- get\_all\_documents: שליפת כל המסמכים מאוסף מסוים.

### 2. Packet

- **תיאור:** מייצגת חבילת רשת עם שדות כמו כתובת מקור, כתובת יעד, פרוטוקול, TTL ועוד. מחלקה זו מספקת מבנה נתונים למידע שנלכד ומספקת פונקציות לסריקת פרוטוקולים (TCP, UDP, ICMP).

- **פונקציות עיקריות:**

- `to_dict`: המרה למבנה מילוני לצורך שמירה במסד הנתונים.
- `extract_icmp_info_`, `extract_udp_info_`, `extract_tcp_info_`: שליפות נתונים ספציפיים מחבילת הרשת.

### Rule .3

- **תיאור:** מייצגת חוק לבדיקת תעבורה נכנסת, כגון כתובת מקור, כתובת יעד, פרוטוקול ופעולה נדרשת.

- **פונקציות עיקריות:**

- `matches`: בודקת האם חבילת רשת תואמת את החוקים המוגדרים במחלקה.
- `raise_alert`: מייצרת התרעה במקרה של חבילה שתואמת את החוק.

### RuleSet .4

- **תיאור:** מייצגת את כללי הניטור במערכת, מנהלת את כלל החוקים, טוענת ומעדכנת אותם מהמסד. פועלת כמחלקת Singleton כדי למנוע ריבוי מופעים של כללי הניטור.

- **פונקציות עיקריות:**

- `add_rule`, `delete_rule`, `edit_rule`: ניהול כללי הניטור והוספתם למסד הנתונים.
- `check_packet`: בדיקת התאמה של חבילה לכלל הכללים המוגדרים.
- `print_all_rules`: הצגת כל החוקים למשתמש.

### AnomalyDetector .5

- **תיאור:** אחראית על זיהוי אנומליות בתעבורת הרשת, למשל חבילות רבות מאותה כתובת IP או תנועה כבדה.

- **פונקציות עיקריות:**

- `check_for_anomalies`: מריצה בדיקה שוטפת לגילוי אנומליות במאגר הנתונים.
- `save_anomaly_result`: שומרת את תוצאות הבדיקה במסד הנתונים.
- `get_anomalies`: שליפת כל האנומליות מהמסד לצורך ניתוח נוסף.

### DBNames .6

- מחלקה זו מחזיקה שמות בסיסי נתונים.

- משמשת להגדרה קבועה של שמות מסדי הנתונים, לדוגמה: NET\_GUARD\_DB.

## **Collections .7**

- מחזיקה את שמות האוספים (Collections) במסד הנתונים.
- כוללת ערכים קבועים כגון PACKETS, RULES, ו-ANOMALIES, המשמשים לארגון הנתונים ב-MongoDB.

## **TYPES .8**

- מגדירה את סוגי השדות ומחזיקה מידע על סוגים מסוימים של ערכים במערכת.
- לדוגמה, INTEGER\_VALUES\_IN\_DB - רשימה של שדות שאמורים להיות מטיפוס מספרי במסד, ו-UPPER\_CASE\_VALUES - שדות שאמורים להיות באותיות גדולות.

## **ERROR\_CODE .9**

- מחזיקה קודים לשגיאות עבור טיפול במצבים בהם אין התאמה לכללים.
- לדוגמה, RULE\_ERROR\_ID = -1 משמש לציון שגיאה כאשר לא נמצאו כללים שתואמים לחבילה.

## **FIELDS .10**

- מחזיקה רשימה של שמות השדות (Fields) הקיימים באוספי המסד.
- לדוגמה, SRC\_IP, DEST\_IP, PROTOCOL וכו'. שמות אלו מסייעים לשמור על עקביות בשמות השדות במערכת ומונעים שגיאות.

## **LABELS .11**

- מחזיקה שמות תוויות (Labels) עבור שדות המידע המוצגים בממשק המשתמש.
- לדוגמה, SRC\_IP כ-"Source IP" וכו'. שימוש בתוויות אלו מקל על תצוגת הנתונים בצורה נוחה ומובנת למשתמש.

## **Ui .12**

- מחזיקה הגדרות הקשורות לממשק המשתמש, כולל HOURS\_BACK, PAGE\_SIZE, ו-CSS עבור מצב כהה.
- הקוד DARK\_MODE\_CSS מספק עיצוב מותאם לממשק במצב כהה, כדי לשפר את חוויית המשתמש.

## 13. Paths

- מחזיקה נתיב (Path) עבור קבצים נדרשים כמו אייקונים.

## 5.3 מערכות היחסים בין המחלקות

### MongoDbClient

- משמשת כ"מחלקת שירות" עבור גישה למסד הנתונים.
- **תלוייה** במחלקות RuleSet ו-AnomalyDetector ומספקת להן גישה ושיטות לניהול ושאלות במסד.
- מקבלת נתונים ממחלקות כגון FIELD, DBNames, Collections, ו-ERROR\_CODE לצורך השימוש במסד.

### Packet

- מחלקה עצמאית המייצגת חבילה בודדת עם פרטי נתוני תעבורה (כגון כתובת מקור ויעד, פרוטוקול).
- מתממשת בעיקר עם RuleSet ו-AnomalyDetector לצורך ניתוח ואיסוף נתוני תעבורה.
- נתמכת בנתונים מתוך FIELD ו-LABELS לצורך יצירת הייצוג הפנימי שלה.

### Rule

- מחלקה המייצגת חוק בודד ומחזיקה פרטי סינון כגון כתובת מקור, יעד, פרוטוקול, ופעולה (כגון "חסום" או "אפשר").
- **משתייכת** ל-RuleSet, וכל מופע של חוק נשמר במערך הכללים ב-RuleSet.
- מתממשת עם Packet לצורך בדיקת התאמה בין חבילה לבין כלל.

### RuleSet

- מחזיקה מערך של כללים (Rule) ומשמשת לבדוק את ההתאמה של כל חבילה מול כללים אלה.
- **תלוייה** ב-MongoDbClient לצורך שמירת כללים וניהולם במסד הנתונים.
- מבצעת בדיקות התאמה מול Packet באמצעות המחלקה Rule.

## **AnomalyDetector**

- מנטרת אנומליות בתעבורה ברשת על ידי בדיקות נתונים ממסד הנתונים.
- **תלויה** ב-MongoDbClient לביצוע שאילתות לגילוי אנומליות ובדיקתן במסד הנתונים.
- מבצעת שאילתות באמצעות פונקציות שמבוססות על Collections ו-FIELDS.

### **DBNames**

- מחזיקה שמות של מסדי נתונים (לדוגמה, NET\_GUARD\_DB) ומספקת קביעות המשמשות את MongoClient.

### **Collections**

- מגדירה את שמות האוספים במסד הנתונים, כגון RULES, PACKETS, ו-ANOMALIES.
- מאפשרת למערכת גישה אחידה ומאורגנת לאוספי המידע השונים.

### **FIELDS**

- מחזיקה את שמות השדות הנפוצים עבור החבילות, הכללים והאנומליות.
- משמשת את כל המחלקות לצורך אחידות בשימוש בשמות השדות.

### **TYPES**

- מגדירה סוגי ערכים במסד הנתונים ומספקת שמות של ערכים ספציפיים.
- נתמכת ב-MongoDbClient לצורך המרת ערכים באופן תקני בעת גישה למסד הנתונים.

### **ERROR\_CODE**

- מחזיקה ערכי שגיאה סטנדרטיים, לדוגמה RULE\_ERROR\_ID, עבור שימוש במערכת.

### **LABELS**

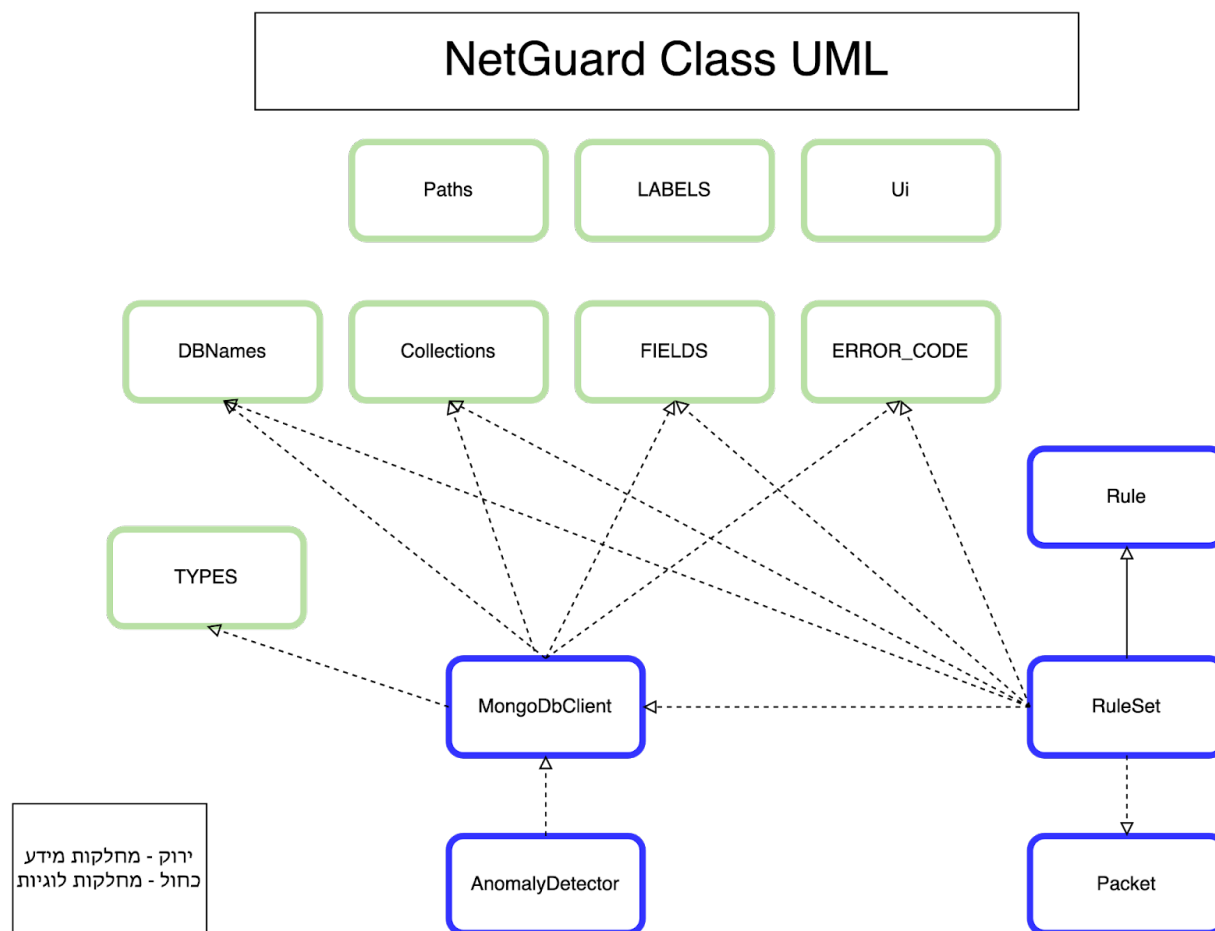
- מחזיקה תוויות להצגת נתוני שדות בממשק המשתמש, כגון Source IP, Destination IP וכו'.

### **Ui**

- מגדירה פרמטרים עבור ממשק המשתמש, כולל מצב כהה והגדרות עיצוב אחרות.

## Paths

- מחזיקה נתיבים עבור קבצים גרפיים, כגון אייקונים, לצורך שימוש בממשק המשתמש.



## 5.4 מחלקות מרכזיות

### מחלקת MongoClient

המחלקה MongoClient אחראית על ביצוע אינטראקציות עם מסד הנתונים MongoDB. היא כוללת מספר מתודולוגיות המיועדות לניהול מסמכים ומבנים ב-MongoDB.

תכונות עיקריות:

1. חיבור למסד הנתונים:

- בעת יצירת מופע של MongoClient, המחלקה מנסה להתחבר ל-MongoDB באמצעות הכתובת 'mongodb://localhost:27017/'. במקרה של כישלון, היא תתפוס את השגיאה ותעדכן את הלוגים.

## 2. שיטות לניהול מסמכים:

- insert\_to\_db: מכניסה מסמך חדש למסד הנתונים ומעדכנת את זמן ההכנסה.
- update\_in\_db: מעדכנת מסמך קיים על פי קריטריונים שניתנים (query) עם נתונים חדשים.
- delete\_from\_db: מוחקת מסמך בהתאם לשאילתת מחיקה.
- get\_data\_by\_field: מחפשת מסמכים בהתאם לשדה ספציפי וערכו.
- find\_by\_id: מחפשת מסמך על פי מזהה ייחודי.
- get\_all\_documents: מחזירה את כל המסמכים מהאוסף المحدد.

## 3. פעולות על אוספים:

- clear\_collection: מנקה את כל המסמכים באוסף שנבחר.
- anomaly\_query: מבצעת שאילתת אגרגציה מותאמת לגילוי אנומליות במסד הנתונים.

## 4. סגירת חיבור:

- close\_connection: סוגרת את החיבור למסד הנתונים.

שימושים:

- המחלקה משמשת כממשק בין האפליקציה למסד הנתונים, מאפשרת לנהל נתונים בצורה פשוטה ויעילה.



## מחלקת RuleSet

המחלקה RuleSet היא singleton שמאגדת חוקים לניהול תעבורת רשת. היא מתמחה בניהול החוקים ובשמירה עליהם במסד הנתונים.

תכונות עיקריות:

### 1. ניהול חוקים:

- בעת יצירת מופע של RuleSet, היא טוענת את כל החוקים ממסד הנתונים ומאחסנת אותם ברשימה.

### 2. תכונות חוקים:

- rules: רשימה של חוקים המיוצגים על ידי אובייקטים מסוג Rule.
- rule\_id\_counter: סופר המזהה את מזהה החוק האחרון שנמצא במסד הנתונים.

### 3. שיטות לניהול חוקים:

- load\_rules\_from\_db: טוענת חוקים ממסד הנתונים לאובייקט.
- add\_rule: מוסיפה חוק חדש לרשימה ולמסד הנתונים, כולל פרמטרים כמו כתובת IP, פורטים, פרוטוקול ופעולה (איפוס או חסימה).

### 4. סנכרון עם מסד הנתונים:

- המחלקה משתמשת ב-MongoDbClient כדי לבצע פעולות כמו הוספת חוקים, טיובם ושימורם.

שימושים:

- מחלקה זו מאפשרת לאפליקציה לנהל בקלות חוקים מורכבים לסינון תעבורת רשת, דבר שמסייע בשיפור אבטחת הרשת.

## 6.0 תהליכים עיקריים

### 6.1 התהליך הראשי

#### תיאור כללי

התהליך הראשי של המערכת NetGuard מאחד מספר פונקציות עיקריות: ניהול כללים, הסנפת תעבורה, זיהוי אנומליות, והפעלת ממשק משתמש. התהליך מוגדר בקובץ ראשי זה, ומריץ מספר Threads במקביל על מנת להבטיח שהמערכת פועלת בצורה יעילה ומהירה.

#### מבנה:

#### ייבוא ספריות:

כל הרכיבים הנחוצים לתהליך מיובאים בתחילת הסקריפט, כולל ניהול כללים, ניהול בסיס נתונים, ניהול תעבורת רשת, ממשק המשתמש והגדרות רישום (logging).

#### פונקציה main():

זוהי הפונקציה הראשית של המערכת, והיא מבצעת את הפעולות הבאות:

#### 1. חיבור לבסיס הנתונים ויצירת אובייקטים:

1. נוצר אובייקט MongoClient להתחברות לבסיס הנתונים.
2. יוצרים אובייקט RuleSet המכיל את כלל החוקים המוגדרים.
3. יוצרים אובייקט AnomalyDetector לניהול ובדיקת אנומליות.

#### 2. הפעלת Threads:

1. **Thread של הסנפת תעבורה נכנסת:** מופעל באמצעות קריאה לפונקציה capture\_packet עם כיוון "IN" ואובייקט החוקים.
2. **Thread של הסנפת תעבורה יוצאת:** מופעל גם הוא באמצעות capture\_packet, אך עם כיוון "OUT".
3. **Thread של זיהוי אנומליות:** מופעל באמצעות קריאה ל-check\_for\_anomalies במטרה לזהות דפוסים חריגים בזמן אמת.

#### 3. הפעלת ממשק המשתמש:

1. ממשק המשתמש מופעל ישירות בתהליך הראשי על ידי קריאה לפונקציה start\_server, אשר מפעילה את ui\_main (שמשתמשת באובייקטים של ניהול חוקים ואנומליות) על פורט 8088.

## 6.1.1 הפעלת ממשק המשתמש מה Thread הראשי

### פונקציה ראשית: ui\_main

- 1. תפקיד:** ה-Thread האחראי להפעלת ממשק המשתמש של NetGuard. הוא מאפשר למשתמשים לצפות בתעבורת הרשת בזמן אמת, לנהל חוקים, לצפות בדשבורד שמציג סטטיסטיקות על תעבורת הרשת, ולנתח את האנומליות שזוהו.
- 2. פרמטרים:**

- 1. rule\_set:** אובייקט המכיל את כל החוקים להגדרת חבילות ואנומליות.
- 2. anomaly\_detector:** אובייקט מסוג AnomalyDetector המאפשר אינטראקציה עם המערכת לצורך בדיקת האנומליות והצגתן למשתמש.

### פונקציות עיקריות

#### פונקציה: show\_packets

- **תפקיד:** פונקציה זו מציגה את כלל הפקטות בזמן האחרון שנכנסו ויצאו מהמחשב. היא מאפשרת לראות את הפקטות, לראות מידע נוסף אודות פקטה נבחרת ולחפש ולפלטר החוצה פקטות לפי פרמטרים שונים.

#### פונקציה: manage\_rules

- **תפקיד:** פונקציה זו מציגה את דף ניהול החוקים בממשק המשתמש, ומאפשרת למשתמש לצפות בחוקים הקיימים, לערוך אותם, למחוק חוקים וליצור חוקים חדשים בהתאם לצרכים.
- **פרמטרים:**
  - rule\_set:** אובייקט שמכיל את כל החוקים המוגדרים במערכת, ומאפשר גישה לניהול ועריכת החוקים.

#### פונקציה: put\_dashboard

- **תפקיד:** פונקציה זו אחראית על הצגת הדאשבורד בממשק המשתמש של המערכת, אשר כולל גרפים וסטטיסטיקות על תעבורת הרשת ועל חוקים תואמים. הדאשבורד מציג את המידע בצורה ויזואלית ובזמן אמת, ומאפשר למשתמש לעקוב אחר נתוני התעבורה, פקטות ומידע נוסף באופן פשוט ונגיש.
- **פרמטרים:**
  - אין. הפונקציה משתמשת בנתונים מהבסיס הנתונים ובפלט שנוצר בפונקציות אחרות.

#### פונקציה: show\_anomalies

- **תפקיד:** פונקציה זו אחראית להצגת כלל האנומליות שהמערכת זיהתה בממשק המשתמש, כולל פרטים חיוניים על כל אנומליה שנמצאה בתעבורת הרשת. המטרה היא להעניק למשתמש יכולת לסקור אירועים חריגים בזמן אמת ולהגיב בהתאם.
- **פרמטרים:**  
anomaly\_detector: אובייקט האחראי לניהול האנומליות ומכיל גישה לנתונים של אנומליות שאותרו.

## 6.1.2 הפעלת ה-Frontend

1. **הפעלת שרת אינטרנט מקומי:** מתחיל בהרצה של שרת אינטרנט מקומי המיועד להציג את ממשק המשתמש. לדוגמה, ניתן להריץ שרת על פורט 8088 המאפשר גישה לממשק בכתובת <http://localhost:8088>.
2. **הרצת פונקציית UI ראשית:** הפונקציה הראשית של ממשק המשתמש (main\_ui) אחראית על בניית דפי הממשק והצגת התכנים באופן דינמי. הפונקציה מקבלת את אובייקטי ה-rule\_set וה-anomaly\_detector כדי להציג נתונים בהתאם לצורך.
3. **שילוב ה-UI עם פונקציות הלוגיקה:** הממשק מקבל את המידע הדרוש מתוך מסד הנתונים ומאפשר הצגה של כל פקטה, חוק או אנומליה שאותרה, כולל אפשרות לאינטראקציות כמו:
  - הצגת כלל הפקטות עם אפשרויות סינון ומידע נוסף על כל פקטה.
  - הצגת טבלת ניהול חוקים עם אפשרויות הוספה, עריכה ומחיקה של חוקים.
  - הצגת דשבורד שמציג סטטיסטיקות על כיוון התעבורה, מספר הפקטות, והפקטות התואמות לכללים.
  - הצגת טבלת האנומליות שהתגלו.

## עקרונות פעולה עיקריים של ה-UI:

- **כפתור מעבר לדפים:** בחלונית הניווט בצד, ניתן לעבור בין עמודים שונים - Packets, Manage Rules, Dashboard, Anomalies.
- **כפתורים לאינטראקציות שונות:** הממשק כולל כפתורים לפעולות שונות כמו Add New Rule, Get Packets ועוד.
- **הצגת סטטיסטיקות ודשבורדים:** הממשק מציג דשבורדים אינטראקטיביים המספקים ניתוח חזותי של תעבורת הרשת וזיהוי תבניות חריגות.

- **טיפול בשגיאות וטעינה מתמשכת:** הממשק מציג למשתמש הודעות שגיאה במידה וקיימות בעיות בהצגת הנתונים או בגישה למסד הנתונים. כמו כן, ישנם אינדיקטורים לטעינה במקרה של עיבוד נתונים מורכב.

### הפעלת ה-Frontend

על מנת להפעיל את ממשק המשתמש כחלק מהמערכת, יש להריץ את הפונקציה הראשית של הממשק main, כך שהלוגיקה של זיהוי האנומליות וההסנפה יכולה לפעול בThreadים מקבילים.

### Thread של הסנפת חבילות (Packet Capture Thread)

#### פונקציה: capture\_packet

- תפקיד: פונקציה זו אחראית על הסנפת חבילות נתונים שנשלחות ומתקבלות ברשת. היא משתמשת ב-Scapy, ספריית פייתון המיועדת לניתוח תעבורה רשתית.
- פרמטרים:
  - direction: מייצג את כיוון החבילה (למשל, נכנסת או יוצאת).
  - rule\_set: אובייקט המכיל את כל הכללים לניתוח החבילות.
- כיצד היא פועלת:
- הפונקציה משתמשת ב-sniff() כדי להאזין לתעבורת ה-IP על הממשק en0.
- היא מעבירה כל חבילה שנלכדת לפונקציה manage\_sniffed\_packet, יחד עם כיוון החבילה והכללים.
- אם מתרחשת שגיאה (למשל, אם לא ניתן לגשת לממשק הרשת), הפונקציה תתעד את השגיאה ביומן.

### Thread של בדיקות אנומליות (Anomaly Detection Thread)

#### מחלקה: AnomalyDetector

- **תפקיד:** מחלקה זו אחראית על זיהוי אנומליות בתעבורת הרשת שנלכדה על ידי הסנפה. היא פועלת ברקע ומבצעת בדיקות סדירות כדי לזהות תבניות חריגות.
- **שיטות עיקריות:**

- **check\_for\_anomalies():**

- פונקציה זו מתבצעת כלולאה אינסופית ומבצעת את הבדיקות האנומליות שנמצאות ברשימה.
- היא בודקת את כל הפונקציות שמייצרות אנומליות (כגון high\_traffic\_on\_any\_port\_anomaly ומביאה את התוצאות מהמאגר (MongoDB).
- אם מזוהה אנומליה, היא מתעדת את התוצאה ב-db.
- מתבצעת המתנה של 10 שניות בין כל מחזור בדיקה, אך ניתן לשנות זאת.

- **save\_anomaly\_result():**

- כאשר אנומליה מזוהה, היא מתועדת ב-db עם פרטים כמו שם האנומליה, זמן גילוי, ותוצאות הבדיקה.

### **הפעלת ה-thread:**

- כדי להפעיל את תהליך בדיקות האנומליות, תצטרך ליצור אובייקט של AnomalyDetector ולהפעיל את השיטה check\_for\_anomalies () כ-thread נפרד. זה יאפשר לתהליך ההסנפה להמשיך לפעול ללא הפרעה בזמן שהאנומליות נבדקות ברקע.

### **Thread של הצגת דאשבורד (Dashboard Display Thread)**

#### **פונקציה: run\_dash\_app**

**תפקיד:** Thread זה אחראי על הצגת דשבורד אינטראקטיבי המציג סטטיסטיקות וגרפים לניתוח תעבורת הרשת. הממשק משלב מגוון כלים וגרפים, המיועדים להמחשת פרטי התעבורה והחוקים הפעילים במערכת.

#### **פרמטרים:**

- אין צורך בפרמטרים נוספים להפעלת הדשבורד עצמו, אך נדרשת גישה לבסיס הנתונים לצורך אחזור נתונים עדכניים לתצוגה.

### **כיצד ה-Dashboard Thread פועל**

- **הפעלת שרת דשבורד עצמאי:** ה-thread מפעיל את הדשבורד על שרת מקומי (בברירת מחדל בכתובת <http://localhost:8050>) באמצעות Dash. דבר זה מאפשר למשתמשים לגשת לממשק הדשבורד בנפרד ולהשתמש בו לצפייה מקבילה במידע בזמן אמת.
- **עדכונים בזמן אמת:** כאשר המשתמש משנה את מספר הימים לתצוגה (לדוגמה, 1 יום, 3 ימים, 7 ימים), הדשבורד מבצע אחזור ועדכון של המידע.
- **שילוב גרפים וסטטיסטיקות:**
  - **כיוון תעבורה (IN vs. OUT)** - מוצג כתרשים פאי המראה את כמות החבילות הנכנסות לעומת היוצאות.
  - **מספר חבילות לפי שעה** - מוצג כגרף קווי המציג את כמות הפקטות שנלכדו לפי שעות היום.
  - **התאמות לפי חוקים** - תרשים פאי המציג את החבילות שהתאימו לכללים מסוימים לעומת חבילות שלא התאימו.
  - **התפלגות חוקים מתאימים** - מציג תרשים פאי עם חלוקה של מספר הפקטות שתאמו לכל חוק באופן ספציפי.

### הפעלת ה-thread

להפעלת ה-thread יש להפעיל את הפונקציה `start_dash_thread()`, היוצרת thread חדש שמריץ את הפונקציה `run_dash_app()` של הדשבורד. הרצה זו מאפשרת המשכיות של פעולת זיהוי אנומליות, הסנפת חבילות וניהול החוקים תוך כדי תצוגת הדשבורד וממשק המשתמש במקביל.

## 7.0 בדיקות

### קובץ ראשון: `handle_network`

#### 1. `test_manage_sniffed_packet`:

##### ○ מה נבדק?

- הבדיקה מוודאת שהתפקוד `manage_sniffed_packet` עובד כמו שצריך.

##### ○ איך זה עובד?

- נבנית חבילה מדמה (mock) של Scapy, עם כתובת מקור וכתובת יעד.

- נבדק אם התפקוד `manage_sniffed_packet` מפעיל את הפונקציה `insert_to_db` של מחלקת `MongoDbClient` כדי לשמור את החבילה.

- **הסבר:** הבדיקה מאמתת שהתפקוד שומר את החבילה במסד הנתונים אחרי שמטפל בה.

#### 2. `test_packet_initialization`:

##### ○ מה נבדק?

- הבדיקה מוודאת שהמחלקה `Packet` מצליחה לאתחל את המידע מהחבילה באופן נכון.

##### ○ איך זה עובד?

- נבנית חבילה מדמה (mock) עם פרמטרים של IP ו-TCP.

- נבדק אם הכתובת של המקור, הכתובת של היעד, והפרוטוקול (TCP במקרה זה) נאספו כראוי.

- **הסבר:** הבדיקה בודקת אם מחלקת `Packet` מזהה ומאחסנת נכון את כל המידע שהחבילה מכילה, כמו כתובת מקור, יעד ופרוטוקול.

#### 3. `test_save_packet`:

##### ○ מה נבדק?

- הבדיקה מוודאת שהתפקוד `save_packet` מצליח לשמור את החבילה במסד הנתונים.

##### ○ איך זה עובד?



- נבנית חבילה מדמה (mock) והפונקציה save\_packet מופעלת.
- לאחר מכן נבדק אם הפונקציה insert\_to\_db הוזמנה כראוי עם הנתונים המתאימים.

○ **הסבר:** הבדיקה מאמתת שהחבילה נשמרת במסד הנתונים דרך פונקציה זו.

## קובץ שני: packet

### 1. test\_packet\_initialization\_tcp

○ **מה נבדק?**

- הבדיקה בודקת אם החבילה שכוללת פרוטוקול TCP מאותחלת נכון במחלקת Packet.

○ **איך זה עובד?**

- נבנית חבילה מדמה עם פרוטוקול TCP ונתוני מקור ויעד.
- לאחר מכן נבדק אם כל המידע (כתובת מקור, כתובת יעד, פורטים, TTL) מאותחל כראוי.

○ **הסבר:** הבדיקה בודקת את פעולתה של מחלקת Packet במקרים של פרוטוקול TCP.

### 2. test\_packet\_initialization\_icmp

○ **מה נבדק?**

- הבדיקה בודקת אם החבילה שכוללת פרוטוקול ICMP מאותחלת נכון במחלקת Packet.

○ **איך זה עובד?**

- נבנית חבילה מדמה עם פרוטוקול ICMP.
- נבדק אם הערכים הנכונים לכתובת המקור, יעד, סוג הודעת ICMP, קוד ועוד מאותחלים כראוי.

○ **הסבר:** הבדיקה בודקת איך מחלקת Packet מתמודדת עם חבילות פרוטוקול ICMP.

### 3. test\_packet\_with\_payload

○ מה נבדק?

- הבדיקה בודקת אם החבילה שכוללת Payload (נתונים נוספים) מאותחלת נכון.

○ איך זה עובד?

- נבנית חבילה מדמה עם פרוטוקול TCP ו-Raw Payload.

- נבדק אם ה-Payload נשמר נכון ובדקו גם את גודל החבילה.

- **הסבר:** הבדיקה בודקת את יכולת מחלקת Packet לאחסן ולהציג את ה-Payload בצורה נכונה.

4. **test\_packet\_fragmentation**

○ מה נבדק?

- הבדיקה בודקת איך מחלקת Packet מתמודדת עם חבילות המפוצלות (fragmented packets).

○ איך זה עובד?

- נבנית חבילה עם דגל "MF" (More Fragments), שהיא חבילה מפוצלת.

- נבדק אם המידע על המקטע (fragment) נשמר כראוי.

- **הסבר:** הבדיקה בודקת את תפקוד מחלקת Packet עם חבילות מפוצלות.

5. **test\_unknown\_protocol**

○ מה נבדק?

- הבדיקה בודקת כיצד מתמודדת מחלקת Packet עם פרוטוקולים לא מוכרים.

○ איך זה עובד?

- נבנית חבילה עם פרוטוקול לא מוכר (פרוטוקול מספר 99).

- נבדק אם המחלקה מחזירה את המידע "Unknown protocol number".

- **הסבר:** הבדיקה בודקת את ההתמודדות עם פרוטוקולים שלא קיימים במפה של מחלקת Packet.

הבדיקות מכסות את כל התרחישים השכיחים שיכולים להתרחש עם חבילות רשת במערכת, מהשגת המידע הבסיסי של החבילה (כמו כתובת מקור ויעד), דרך עבודה עם פרוטוקולים שונים (TCP, ICMP), ועד להתמודדות עם חבילות מפוצלות ופרוטוקולים לא מוכרים.

## 8.0 רעיונות לשיפור והרחבת המערכת

להלן נפרט ונציע רעיונות לפיתוחים עתידיים ושיפורים למערכת NetGuard, אשר יכולים להרחיב את יכולות המערכת ולשפר את ביצועיה בתחום ניטור ואבטחת התעבורה. השיפורים המוצעים כוללים תוספות פונקציונליות, שדרוגי UI, והעמקת אנליזות ויכולות זיהוי, שיכולים לתרום משמעותית ליכולות המערכת ולחווית המשתמש.

### 1. הוספת סטטיסטיקות ויזואליות נוספות לדשבורד:

- **פילוח תעבורה לפי פרוטוקול** - הוספת גרפים המפרטים את סוגי הפרוטוקולים (TCP, UDP, ICMP) והשכיחות שלהם בתעבורה.
- **מעקב אחרי רוחב הפס** - הוספת גרפים הממחישים את השימוש ברוחב הפס לפי פרקי זמן שונים, כדי לזהות עומסים.
- **זיהוי מגמות לפי שעה ויום** - הוספת סטטיסטיקות המציגות מגמות תעבורה לפי שעות ביום או ימים בשבוע, המסייעות בזיהוי דפוסי פעילות חריגים.

### 2. הוספת אנומליות וניטורים נוספים:

- **ניטור ניסיונות חיבור מופרזים לפורטים סגורים** - זיהוי של פעולות Scanning על המערכת וניטור ניסיונות חיבור לפורטים שאינם בשימוש.
- **זיהוי תבניות תנועה חריגות** - פיתוח אלגוריתמים שמזהים שינויים פתאומיים בתבניות התנועה, כמו עלייה חדה במספר הפניות לפורט ספציפי או שינוי בתדירות הבקשות מפרוטוקולים שונים. ניטור זה יכול לעזור בזיהוי מתקפות כמו DDoS.
- **ניטור פעילות משתמש חשודה** - הוספת יכולת לניטור פעילות חשודה ברמת המשתמש, כמו מספר ניסיונות גישה לא מוצלחים בפרק זמן קצר, או גישה למערכות מחוץ לשעות הפעילות. אפשרות זו תוכל לסייע בזיהוי ניסיונות פריצה או התחזות מתוך הרשת.

### 3. הרחבת חוקים וגמישות בתנאים:

- **חוקים מבוססי תוכן** - הגדרת חוקים לזיהוי תוכן מסוים בתוך הפקטות, מה שיכול לשמש לסינון תוכן רגיש או מסוכן.
- **חוקים מותאמים מבוססי זמן** - קביעת חוקים שיחולו בזמנים ספציפיים ביום או בהתאם לדפוסים מסוימים.
- **חוקים דינמיים** - הוספת אפשרות למערכת לעדכן את החוקים בצורה אוטומטית על בסיס נתוני התעבורה והאנומליות הקיימות.

### 4. חסימת פקטות ברמת הקרנל:

- **אינטגרציה עם כלים כמו Netfilter** - שימוש במודולים קיימים בקרנל, כמו Netfilter, שיאפשרו טיפול וניהול תעבורה בצורה אפקטיבית עוד ברמת הליבה של המערכת.

### 5. מערכת התראות ותגובה חכמה:

- **מערכת התראות לפי חומרת האירוע** - הצגת התראות צבעוניות על בסיס חומרת האנומליה או האירוע שזוהה.
- **פעולות תגובה אוטומטיות** - הוספת כלים המאפשרים למערכת לבצע תגובות אוטומטיות לאירועים מסוימים, כגון ניתוק חיבורים חשודים או הקשחת חוקים במקרה של זיהוי מתקפה.

### 6. התאמה למערכות מורכבות ולתעבורה גבוהה:

- **אופטימיזציה לניהול תעבורה גבוהה** - התאמה של המערכת לעבודה בסביבות בעלות תעבורה גבוהה תוך שמירה על ביצועים גבוהים.
- **התממשקות עם מערכות קיימות** - אפשרות להתממשק עם מערכות ניטור ואבטחה חיצוניות לצורך ניתוחים מתקדמים..

### 7. בדיקות מורכבות ונוספות:

- ניתן להוסיף עוד בדיקות לכל חלקי המערכת ולבצע בדיקות עומס על המערכת.

## 9.0 סיכום פרויקט NetGuard

פרויקט NetGuard פותח במטרה לספק מערכת חכמה לניטור ואבטחת תעבורת רשת עבור ארגונים ומשתמשים פרטיים. המערכת מאפשרת למשתמשים לנטר ולנהל את תעבורת הנתונים העוברת במחשבים וברשתות שלהם, לזהות פעולות חשודות בזמן אמת ולהגיב אליהן במהירות.

### המערכת כוללת יכולות כמו:

1. **ניהול חוקים מותאמים אישית** – אפשרות ליצור, לערוך ולמחוק חוקים המגבילים או מאפשרים סוגי תעבורה מסוימים, יחד עם קבלת התראות על תעבורה חריגה.
2. **זיהוי אנומליות ואיומים** – מנגנון לזיהוי דפוסים חשודים בתעבורת רשת כמו התקפות DDoS, פריצות וניסיונות התחברות מוגזמים.
3. **Dashboard גרפי** – לוח בקרה אינטראקטיבי המאפשר למשתמשים לצפות בסטטיסטיקות חיות על תעבורת הרשת ולהבין את אופי התעבורה בקלות.
4. **ניהול פקטות** – מממשק משתמש לצפייה בפקטות, חיפוש, סינון וניתוח פקטות בצורה מפורטת.

המערכת פותחה בשפת python באמצעות טכנולוגיות MongoDB, Scapy, Dash, PyWebIO ומבוססת על ארכיטקטורה מודולרית המאפשרת הרחבות ושיפורים עתידיים.

### יעדים עיקריים שהושגו בפרויקט:

- ניטור תעבורת רשת והתראות בזמן אמת.
- ניהול חוקים להגבלת תעבורה לפי קריטריונים מותאמים אישית.
- איתור אנומליות בנפחי תעבורה ודפוסים חריגים.
- הצגת דשבורד וסטטיסטיקות לתמיכה בהחלטות של מנהלי אבטחת המידע.

פרויקט NetGuard מספק כלי רב עוצמה לניהול ואבטחת רשת, ומציג פתרון חדשני לצרכים ההולכים וגוברים בתחום אבטחת המידע.