עבודת גמר 5 יח"ל

נושא העבודה : Packet Tracer רשתי ללימוד רשתות וסייבר

שם תלמיד :

ת.ז תלמיד :

שם בית ספר ועיר : קריית החינוך ע"ש עמוס דה-שליט, רחובות

שם המנחה : ערן בינט

מועד הגשה : 28/05/2022

תוכן עניינים

1. מבוא

1.1. נושא העבודה

נושא העבודה שלי הוא אמולטור רשתי המדמה Packet Tracer בדגש על שכלול הבנת נושא הרשתות. המערכת תדמה רשת אמתית דרך יצירת רכיבי תקשורת מדומים (כגון: נתב, מחשב ומתג) ושליחת מידע מנקודות קצה לנקודות קצה דרך רכיבי התקשורת המדומים ברשת בהתאם לפרוטוקולי התקשורת השונים.

המערכת מאפשרת למנהל הרשת (ADMIN) לצפות ברכיבי הרשת השונים הקיימים ברשת, בפרטיהם, בכתובתם (פיזית ולוגית) ובפרטי הרשת (וה- subnet) אליה הם משתייכים. המערכת תציג את התקשורת העוברת ברשת המדומה באופן גרפי, תוך הצגה וניתוח בסיסי של התעבורה (אנימציה של החבילות העוברות ברשת, סוג הפרוטוקול והמידע העובר בחבילות) . עוד תאפשר המערכת ניהול ועריכה של הרשת – הוספה ומחיקת רכיבים מהרשת, הגדרה של רכיבים (לדוגמה routing table סטטית ודינאמית) ושליחת פקטות בין נקודות קצה על בסיס פרוטוקולי תקשורת שונים (למשל udp,tcp, ping ועוד).

התוכנה עצמה רצה על מחשבים שונים שכל אחד מהם מתפקד כנתב בתוכנה, ואלה מתקשרים עם ה SERVER שמתקשר עם ה ADMIN באמצעות SOCKET. המערכת מתבססת על שימוש מרכזי בכלי Scapy – המאפשר שליחה וקבלת פקטות (הסנפה) הנוצרות באופן ידני ע"י המשתמש, באמצעותו ניתן לממש פרוטוקולי תקשורת ולקיים תקשורת בין הרכיבים השונים ברשת המדומה.

1.2. מטרות מרכזיות

המטרות היישומיות המרכזיות של העבודה הן:

- ניהול מרכזי של מנהל הרשת הכולל יצירת רכיבים ברשת והגדרתם (חוץ מיצירת נתבים שייווצרו כברירת מחדל כאשר לקוח מתחבר לשרת), בהתאם לכך תתאפשר יצירה והגדרה של רכיבי תקשורת שונים כגון מחשבים ומתגים.
- הצגה גרפית של תקשורת אמיתית (כזאת שניתן להסניף ב Wireshark), מעל פרוטוקולים ברמות 2-4 של מודל OSI.
- בניית גרפיקה מרשימה ונוחה לשימוש המשתמש.

המטרות האישיות המרכזיות של הפרויקט:

- למידה של רשתות ברמה גבוהה – הבנה עמוקה של פרוטוקולי תקשורת ותפקיד רכיבי תקשורת.
- התמקצעות ב Winforms - עיצוב גרפי יסודי ויפה, ושליטה בפונקציונליות של הכלי.
- לימוד תבניות של התקפות סייבר – זיהוי התקפות ברשת והבנת ההיגיון מאחוריהן לעומק.
- יישום הפרויקט ברמה גבוהה, על בסיס התכנון המשתקף ב PRD, באופן הממצה את יכולותיי כתלמיד במגמת סייבר.
- עמידה בלוח זמנים שקבעתי עם עצמי (וכמובן עם המנחה) מראש.

- תוצר מרשים בעל אפשרויות מגוונות למשתמש, הן ברמת הצגת ניתוח התעבורה והן ברמת יישום פרוטוקולי תקשורת שונים אופציונליים (כגון: TLS,DNS,HTTP,SMTP).

1.3. רציונל

המוטיבציה שלי לפיתוח הפרויקט הינה ליצור כלי ידידותי למשתמש וקל לשימוש אשר יאפשר למידה והתנסות עם רשתות ורכיבי תקשורת בצורה ויזואלית. בעזרת השליטה המוחלטת של המשתמש על הרשת ובעזרת ייצוגה הגרפי, תתאפשר הבנה עמוקה של התיאוריה המופשטת של רשתות תקשורת (פרוטוקולי תקשורת ורכיבי תקשורת). בניגוד לכלים הקיימים בשוק, המוצר שלי יאפשר למידה יותר ממוקדת של רשתות גם לאנשים עם ידע מועט בתחום התקשורת ויחדש בכך שיאפשר יצירה של תקשורת אמתית, שאותה אפשר לראות בעזרת כלים כמו Wireshark. תקשורת זו תיתן למשתמש ממד מוחשי לתחום הרשתות ותעזור לו להבין לעומק איך מתבצעת בפועל תקשורת בין רכיבי רשת שונים. אני חושב שהעולם שלנו נע יותר ויותר לשמירת מידע באינטרנט, ולכן חשוב ללמוד את נושא הרשתות, ולדעת ידע בסיסי בנושא התקפות והגנת סייבר.

באמצעות הפרויקט אני שואף להגיע לרמת ידע גבוהה בנושא רשתות, הן ברמת הרשת הפרטית (LAN) והן ברמת ה Web - להבין לעומק את הפרוטוקולים ותהליכים המתרחשים ברשת, כל זאת דרך יישום של אותם פרוטוקולי ורכיבי תקשורת. בנוסף על כך, אני מצפה לצבור ניסיון בפיתוח ויצירת פרויקט בסדר גודל כזה.

1.4. קישור לחומר הנלמד

העבודה מתקשרת לחומר הנלמד במספר תחומים שונים.

ראשית, לתחום התקשורת והרשתות. בכיתה י"א, רשתות ותקשורת היו הנושא המרכזי בקורס. יישום הפרויקט מתבסס על התיאוריה שלמדנו בשיעורים אך דורש הרחבה וחקירה עצמית בבית. התקשורת בין ה ADMIN, המשמש כממשק המשתמש למערכת, לבין השרת, ודרכו ל clients (הנתבים) השונים, מתבצעת באמצעות Sockets ברמת ה IP/TCP – רמה 4 במודל 7 השכבות. בנוסף באמצעות scapy אני בונה את פרוטוקולי התקשורת עליהם למדנו בכיתה, זאת באמצעות שליחה, האזנה ועריכת פקטות.

תחום נוסף בו הפרויקט שלי נוגע הוא תחום הגרפיקה. במסגרת הפרויקט איישם גרפיקה מורכבת ב WinForms C#. בשנת י"ב למדנו בתחילת השנה כמה מצגות על הנושא. גם פה תידרש חקירה מעמיקה יותר מהרמה שנלמדה בכיתה ומחשבה על עיצוב גרפי הולם.

2. תיאוריה

2.1. תיאוריה

הפרויקט עוסק בתחום התקשורת ובניית רשתות. נהוג לנתח תקשורת בין מחשבים על פי מודל שבע השכבות של ISO המכונה OSI (Open System Architecture). המודל מציג את הפעולות השונות הדרושות על מנת להעביר נתונים ברשת תקשורת, לפיו מידע הנשלח ברשת מקבל ביטוי שונה בחלוקה ל 7 רמות תקשורת, כל אחת בעלת תפקיד מוגדר.

המודל פועל בשני כיוונים, בכיוון הצד השולח את התקשורת וגם בכיוון הצד המקבל את התקשורת. הצד השולח בונה את המידע מהרמה השביעית עד לרמה הראשונה, כלומר את הנתונים שהמשתמש רוצה להעביר (השכבה השביעית – רמת היישום), דרך כל רמות התקשורת שמטרת כל אחת מהן היא לבצע פעולות על הנתונים, בהסתמך על הנתונים והשינויים שהרמות הקודמות לה ביצעו, עד לשכבה הראשונה (השכבה הפיזית – המתבטאת כביטים הנשלחים כגלים או דרך כבלים). תהליך זה נקרא "כימוס" – כל פעם שהנתונים יורדים שכבה מתווספים אליהם עוד שינויים הכרחיים, עד לכדי שליחת המידע בצורה שהצד השני יוכל לקרוא אותה. התהליך ההפוך נקרא תהליך "קילוף", אותו מבצע הצד המקבל את המידע. בתהליך ה"קילוף" הצד המקבל מקלף את המידע החל מהשכבה הראשונה עד לשכבה השביעית, כך שעל כל פעולה שהשכבה המקבילה (שכבה ראשונה מול ראשונה וכן הלאה) בתהליך ה"כימוס" עשתה על המידע, השכבה בצד המקבל יודעת "לקלף" אותה לכדי המידע שהיה בשכבה הקודמת. הצד המקבל "מקלף" את המידע עד שמגיע המידע שהמשתמש השולח רצה להעביר.

רשת מקומית LAN (Local Area Network), למשל פועלת ברמה השנייה במודל ה OSI. רשת זו הינה רשת שבנויה מקבוצת מחשבים ורכיבי תקשורת המחוברים זה לזה בקו תקשורת משותף (חוטי או אלחוטי Wi-Fi). המערכת אותה בונה המשתמש בפרויקט זה מורכבת מאחת או יותר רשתות מקומיות המקושרות זו לזו. את רשתות אלו מרכיבים שלושה רכיבי תקשורת:

- מחשב – רכיב תקשורת שתפקידו, בכל הנוגע לרשת, הוא להוות נקודת קצה לתקשורת העוברת ברשת, כלומר, הוא יכול לשלוח או לקבל חבילות מידע. למחשב יש כתובת פיסית קבועה הקרויה כתובת MAC, המשמשת ל"זיהוי" כרטיס הרשת במחשב השולח ומחשב היעד ברמה השנייה במודל ה OSI. בנוסף, למחשב יש גם כתובת לוגית, המכונה כתובת IP, המזהה באופן חד-חד ערכי מחשבים ב web, כמתחייב ברמה השלישית של מודל ה OSI.
- מתג – רכיב תקשורת שתפקידו לחבר בין רכיבים שונים ברשת. המתג מיוחס לרמה השנייה במודל OSI מכיוון שהוא מעביר את התקשורת באופן סלקטיבי לפי כתובות ה MAC של היעד המבוקש. המתג מתחזק טבלת CAM (Content Addressable Memory) המקשרת בין הכתובות הפיסיות של רכיבי התקשורת לבין ה ports במתג, המובילים אליהם. כאשר חבילה מגיעה אל המתג, המתג בודק מה היציאה המתאימה לכתובת ה MAC אליה מיועדת החבילה: אם הכתובת מוכרת למתג בטבלת ה MAC הוא יעביר את החבילה אך ורק דרך היציאה שתוביל את החבילה ליעדה, אחרת, המתג יעביר את החבילה דרך כל היציאות פרט לזו שממנה היא התקבלה.

- נתב – רכיב תקשורת הנועד לקביעת "נתיבן" של חבילות ברשת ולהפצתן אל מחוץ לרשת המקומית, תפקיד המיוחס לשכבה השלישית במודל ה OSI. הנתב מתחזק טבלת ניתוב RIB – (Routing Information Base), המכילה כתובות יעד של רשתות (Sub-Network), את ה"מסכה" של הרשת (Subnet Mask), את כתובת הקפיצה הבאה כדי להגיע ליעד (interface), ואת הציון (metric) של אותו נתיב, שבמקרה של הפרויקט שלי מהווה את מספר הנתבים שחבילת המידע תעבור דרכם, כדי להגיע לכתובת יעד.

דרך קביעת הניתוב יכולה להתבצע בשתי דרכים – סטטי או דינאמי. גישת הניתוב הסטטי קובעת כי המשתמש קובע נתיב (שורה בטבלת הניתוב) שבו תנותב החבילה לפי העדפותיו. בגישת הניתוב הדינאמי, הנתב משתמש בפרוטוקול תקשורת, כגון RIP, Routing (Information Protocol) הקובע את תחילת הנתיב למציאת Subnet. כחלק ממשימת הניתוב הדינאמי, עשוי הנתב לספור את מספר הנתבים בהם עוברת החבילה עד להגעתה ליעדה . בשיטה זו מנסה הנתב לחשב את מספר הנתבים שעשויה חבילת המידע שברשותו לעשות בדרך ליעדה. אם קיימים כמה מסלולים אפשריים ליעד, עשוי הנתב לבחור את המסלול בו מספר הנתבים שהוא עובר עד להגעתו לרשת היעד הוא הקטן ביותר, ובכך באופן תאורטי יחסך חלק מזמן המסע של החבילה.

רכיבי תקשורת ברשת מתקשרים בעזרת פרוטוקולי תקשורת המותאמים למודל ה OSI. על מנת ליישם מערכת מדמה מציאות, הפרויקט עושה שימוש במספר פרוטוקולים, כמפורט:

- פרוטוקול ARP (Address Resolution Protocol), המשמש לאיתור כתובת MAC של רכיב ברשת לפי כתובת הלוגית שלו (IP). הפרוטוקול מיושם כ payload של הרמה השנייה במודל ה OSI, על מנת לאתר את כתובת ה MAC הפיסית התואמת לכתובת ה IP הלוגית של הרכיב הבא בנתיב. איתור הכתובת הפיסית מתבצעת על ידי שידור חבילת הפצה (broadcast), המכונה בקשת ARP, המתייחסת לכתובת הלוגית של היעד תוך ציון כתובת broadcast "FF:FF:FF:FF:FF:FF " בשדה היעד. הרכיב שיזהה את כתובת ה-IP שלו בתוכן החבילה, יענה בתשובת ARP ישירות אל רכיב המקור, תוך ציון כתובת ה MAC שלו.

- פרוטוקול TCP (Transmission Control Protocol), המשמש להעברת מידע בצורה מנוהלת בין שתי נקודות קצה ברשת. הפרוטוקול פועל ברמה הרביעית במודל ה OSI. TCP מבטיח העברה אמינה ומנוהלת של נתונים בין שתי נקודות קצה ברשת באמצעות יצירת חיבור מקושר. בעת הקמת הקשר בין שני מחשבים, משתמש הפרוטוקול בלחיצת יד בשלושה שלבים (three-way handshake):
  - SYN: מחשב א' שולח הודעה לפתיחת קשר (הודעה בפרוטוקול TCP בה דגל ה-SYN בפתיח נושא ערך "1").

- SYN-ACK: מחשב ב' מקבל את ההודעה ושולח בתגובה הודעת אישור קבלה ואישור פתיחת קשר מצדו (הודעה בפרוטוקול TCP בה דגלי ה- SYN וה- ACK בפתיח נושאים ערך "1").

- ACK: מחשב א' מיידע את מחשב ב' על סיום "לחיצת הידיים" בהודעת ACK (הודעה בפרוטוקול TCP בה דגל ה- ACK בפתיח נושא ערך "1").

לאחר סיום לחיצת הידיים, מצייני ה Seq (שמטרתה לעקוב אחרי מספור החבילות שהתקבלו) וה Ack (שמטרתה לעקוב אחרי מספור החבילות שמחכות לאישור קבלה) הם 1 ו 1. שאר התקשורת מתבצעת בדרך הבאה - צד א' שולח את מספר הבתים שיש בה (Length או בקיצור Len). למשל: Seq=1 Len=3. במקביל מפעיל צד א' שעון עצר, ואם לא מתקבל אישור מהצד השני על קבלת החבילה עד פקיעת השעון הוא שולח שוב את חבילת המידע. צד ב' בתגובה שולח חבילת אישור כתגובה (עם דגל ACK מודלק) עם ערך Ack=4. כך, צד ב' מציין שהוא קיבל את החבילה הקודמת שנשלחה, ומוכן לקבלת החבילה הבאה אחריה. כך מתבצע התהליך עד לסיום הקשר.

- פרוטוקול UDP (User Datagram Protocol), המאפשר העברת נתונים לא מנוהלת בין שתי נקודות קצה ברשת. פרוטוקול UDP פועל ברמה הרביעית במודל ה OSI. הפרוטוקול הוא "connectionless", כלומר אינו יוצר קשר מנוהל (לדוגמה "לחיצת ידיים" בפרוטוקול TCP) בין נקודות הקצה, ולכן אינו מבטיח העברה אמינה של נתונים בין שתי נקודות קצה. גישה זו אמנם יכולה לגרום לאיבוד חבילות או להגעה של חבילות בסדר שונה ממה שהיו אמורות להגיע, אך גורמת לכך שהשימוש בפרוטוקול מאפשר תקשורת מהירה יותר. המהירות שלו הופכות אותו מתאים למקרים בהם אין חשיבות גבוהה לאמינות המידע ויש הרבה מידע להעביר, לדוגמה כשאתר רוצה להציג סרטון.

- פרוטוקול ICMP (Internet Control Message Protocol), המשמש לשליחת הודעות שגיאה במערכת, לצורכי בדיקה של מקרי קצה, או לצורך דווח למערכת על מקרים בלתי צפויים. פרוטוקול ICMP נועד כדי לתווך בין הרמה השלישית והרביעית של מודל ה OSI. במערכת שלי קיימים שלושה סוגי הודעות ICMP:
  - icmp echo – כאשר מחשב רוצה לבדוק אם ניתן ליצור תקשורת עם מחשב אחר, אותו מחשב שולח למחשב האחר חבילת icmp echo, אם מתקבלת חבילת icmp echo תשובה ממחשב היעד, סימן שניתן לבצע תקשורת.
  - icmp unreachable – כאשר לא ניתן להגיע לרשת מסוימת או למחשב מסוים ברשת, הנתב שולח בחזרה למחשב השולח חבילת icmp unreachable המיידעת אותו שלא ניתן לגשת לאותה רשת או מחשב.

- icmp redirect – אם נוצר מצב שנתב שדרכו מנותבת חבילה רואה שלו יש דרך קצרה יותר להגיע לנקודת קצה היעד, שולח הנתב חבילת icmp redirect לנתב ממנו התקבלה החבילה, עם כתובת הנתב שממנו ניתן לקצר את הדרך.

- פרוטוקול RIP (Routing Information Protocol), שהינו פרוטוקול הניתוב הדינאמי הראשון שנוצר עבור ה ARPANET. פרוטוקול RIP פועל בשכבה השלישית במודל ה OSI. נתב המשתמש בפרוטוקול RIP, מנהל רישום של כל הנתבים אותם הוא "מכיר", הרשתות המחוברות אליהן, וכמות הנתבים אותם צריכה חבילה לעבור בכל נתיב לכל יעד (שורה בטבלת ניתוב). כל זמן מוגדר מראש, כל הנתבים שולחים חבילת RIP לכל הנתבים אותם הם "מכירים", המכילה את כל רשומות הניתוב בטבלת שלהם. הנתבים מקבלים את החבילות, ובודקים על כל אחת מהרשומות בחבילה אם הם "מכירים" את רשת היעד המצוינת ברשומה. אם לא – הם מוסיפים אותה לטבלה. אם כן, בודקים אם הנתיב המצוין ברשומה יותר יעיל (לדוגמא, על פי מספר נתבים קטן יותר) לניתוב חבילה ליעד מסוים מאשר הנתיב המצוין בטבלת הניתוב, אם אכן הנתיב המצוי ברשומה יותר יעיל, הנתב יחליף את הרשומה הקיימת בטבלת הניתוב לרשומה מהחבילה. כך כל הנתבים ברשת מתעדכנים על שינויים במערכת ומייצרים ניתוב דינאמי.

2.2. מוצרים קיימים

מערכת דומה הקיימת היום בשוק היא Cisco Packet Tracer:
https://www.netacad.com/courses/packet-tracer

Cisco Packet Trace היא תוכנה המדמה רשת מודרנית. התוכנה מאפשרת למשתמש ליצור רשת פרטית (לא מחוברת ל WEB) המכילה רכיבים מסוגים שונים (של cisco), כדוגמת כבלים, מתגים, רכזות (HUB) ונתבים. התוכנה מיועדת לתכנון רשתות ביתיות ומשרדיות (חברות רבות משתמשות בזה), ומספקת הצגה ויזואלית אסטטית של רכיבי הרשת והתעבורה בה וניהול גרפי נוח המאפשר להגדיר רכיבי תקשורת שונים והגדרת לפי רצון המשתמש. התוכנה תומכת בפרוטוקולי תקשורת מגוונים ברמות 2-4 במודל שבע השכבות, ומאפשרת לימוד רשתות ברמה אקדמית.

הייחוד של המערכת שלי הוא שהמערכת משתמשת בפקטות אמתיות הנבנות עם scapy, דבר המאפשר להסניף את התעבורה בתוכנת Wireshark המשמשת גם כתוכנת לימוד של רשתות. יצירת תקשורת אמתית ברת הסנפה הינה יתרון משמעותי בתחום לימודי הסייבר, מכיוון שהיא מאפשרת מאפשרת באופן ייחודי לימוד על סוגים שונים של פרוטוקולי תקשורת וניתוח לעומק של התעבורה. בנוסף, המערכת שלי היא פשוטה וקלה להבנה גם לאנשים עם ידע מועט בתקשורת, זאת בניגוד למערכת של cisco הדורשת הבנה מקיפה בתיאוריה של רשתות ובסוגים שונים של רכיבי תקשורת וכבלים הקיימים בשוק.

ניתן לייצג את ההבדלים בטבלה:

תבחינים\ פרויקט	הפרויקט שלי	Cisco Packet Trace
סימולציה של רשת	מוצגת למשתמש תצוגה גרפית של רשת וירטואלית	מוצגת למשתמש תצוגה גרפית של רשת וירטואלית
יכולת הסנפת הפקטות (שימוש בפקטות אמתיות)	ניתן להסניף את התקשורת בעזרת כלים כמו Wireshark וScapy	לא ניתן להסניף את התקשורת
אפשרות להגדרה של כבלים, נתבים, מחשבים ומתגים מסוגים שונים ברשת הווירטואלית	לא ניתן	ניתן. בעזרת הגדרת רכיבים מסוגים שונים ניתן לבחון כיצד הם מתפקדים ברשת אמתית
רץ על מחשב (או מכונה וירטואלית) יחיד	מצריך יותר מיחידה אחת של מחשב/מכונה וירטואלית דבר המקשה על הקמת המערכת	לא מצריך יותר מיחידה אחת של מחשב/מכונה וירטואלית דבר המקל על הקמת המערכת.
הצגת חלון הודעות RIP	האופציה קיימת	האופציה לא קיימת
הצגת התקשורת ברשת	קיימת הצגת התעבורה ברשת הווירטואלית	קיימת הצגת התעבורה ברשת הווירטואלית
דרושה הרשמה הכוללת פרטים אישיים	לא דרושה הרשמה הכוללת פרטים אישיים, דבר השומר על אנונימיות המשתמש	דרושה הרשמה ויצירה של משתמש עם פרטים אישיים , דבר החושף את פרטיו האישיים של המשתמש בפני חברת cisco ובכך את המשתמש לקבלת הודעות spam ופרסומות
נדרש תשלום	לא נדרש לשלם על האפליקציה	נדרש לשלם על האפליקציה
אפשרות להגדרה של יותר ממתג אחד	לא ניתן להגדיר יותר ממתג אחד במערכת.	ניתן להגדיר מספר לא מוגבל של נתבים.
הגבלה של מספר רשתות במערכת	ניתן להגדיר עד שש רשתות במערכת.	ניתן להגדיר מספר לא מוגבל של רשתות במערכת.

3. תוצר סופי

3.1. תיאור הפרויקט

המערכת הינה מערכת מסוג client-server, אשר המשתמש בה הוא ה Admin. רכיבי המערכת הינם:

- שרת – תוכנה המנהלת את המערכת, שולטת על הגדרה רכיבים ותקשורת בין הרשתות.
- לקוח – תוכנה שרצה במחשב פיזי שתפקידה להגדיר רשת מקומית בטופולוגיה הרשתית, ומנהלת את התקשורת מול השרת.
- נתב – תוכנה, שרצה במחשב הלקוח, המייצגת נתב ברשת הווירטואלית. הנתב נוצר אוטומטית על ידי המערכת בעת צירוף רשת מקומית נוספת למערכת.
- מחשב – תוכנה שרצה במחשב הלקוח, המייצגת מחשב ברשת הווירטואלית. ייווצר לפי הוראת המנהל.
- מתג - תוכנה שרצה במחשב הלקוח, המייצגת מחשב ברשת הווירטואלית. ייווצר לפי הוראת המנהל.

כאשר תעלה המערכת, המשתמש (ה ADMIN) יקבל תצוגה של כל הרשתות המקומיות המחוברות, על פי תמונת המחשבים בהם מופעל client. המשתמש יוכל לבנות רשתות כרצונו, להוסיף ולחבר רכיבי תקשורת (מחשבים ומתגים), וכן להגדיר טבלת ניתוב באופן ידני או דינאמי (כלומר לקשר את אותן רשתות המקומיות המחוברות למערכת).

המשתמש יוכל ליזום פקטות מעל פרוטוקולי תקשורת מגוונים ולהגדיר את המידע המועבר. המערכת תציג בצורה גרפית ברורה את התעבורה בין רכיבי התקשורת השונים ברשתות השונות, ואת האישור לגבי הגעתה ליעד. המערכת תאפשר ניתוח בסיסי של התעבורה הכולל לוג מתעדכן של התעבורה ברשת ותצוגה של כל שלבי התקשורת בין נקודות קצה שונות בצורה גרפית. כל רכיב רשת שיוגדר יופיע בחלון הגרפי הראשי של האפליקציה, בנוסף לתכונות בסיסיות שלו (למשל MAC ו IP). יהיה ניתן ללחוץ פעמיים על כל רכיב ולדעת פרטים יותר מורכבים, למשל, לאיזה רכיבי תקשורת הוא מחובר, טבלאות למיניהם (תלוי ברכיב, למשל לנתב יוצג טבלת ניתוב) ולאיזה SUBNET הוא שייך.

ממשק המשתמש יהיה ברור ויאפשר התנהלות פשוטה, כך שהשימוש במערכת יהיה אינטואיטיבי בתנאי ידע בסיסי ברשתות ורכיבי תקשורת. המערכת תדע לבצע:

- הוספה של רכיבי רשת שונים, הגדרתם וקישורם לרכיבי תקשורת אחרים
- צפייה בפרטיהם של רכיבי הרשת ובתעבורה
- ביצוע תקשורת בין רכיבי תקשורת, באופן מנוהל על ידי המשתמש בעזרת שורת פקודה (CMD).
- ניתוח בסיסי של התעבורה הכולל לוג של התקשורת הכולל ניהול שגיאות.

- תצוגה גרפית של התקשורת בין נקודות קצה שונות עם אפשרות של המשתמש לצפות בתהליך התקשורת שלב אחר שלב.

3.2. אלגוריתמים עיקריים

אלגוריתמים מרכזיים במערכת:

1. שליחה וקבלת פקטות בין רכיבי קצה ברשת – בעזרת Scapy:

התוכנית בונה ושולחת פקטות באמצעות Scapy, על בסיס מידע שנותן המשתמש (mac, ip, dest, protocol). הרכיבים הרלוונטיים מסניפים את הפקטות שמיועדות להם, הן ברמת ה IP והן ברמת ה MAC. תוכנת רכיב התקשורת מנתחת את הפקטות שהוסנפו, ואם פרוטוקול התקשורת דורש זאת, מייצרת תקשורת המשכית (למשל אם נשלח ICMP ECHO אז הרכיב המקבל את ההודעה ישלח הודעת ICMP כתגובה).

כאשר המשתמש בוחר לבצע תקשורת, כלומר לשלוח פקטה מרכיב A לרכיב B:

- השרת שולח ללקוח בו נמצא רכיב A, הוראה ליצור פקטה.
- הלקוח כותב את הפקטה בקובץ TXT.
- כאמצעות LISTENER, רכיב התקשורת A קורא את הפקטה מהקובץ.
- A יוצר פקטה בעזרת ספריית scapy עם הפרטים שנקראו מהקובץ ושולח אותה לנתב במחשב הלקוח (אם היעד מחוץ לרשת הפרטית) וישירות לרכיב B, אם היעד בתוך הרשת הפרטית.
- אם B מחוץ לרשת הפרטית, הנתב מנתב את הפקטה אל הרשת הפרטית בה B נמצא (כמפורט באלגוריתם אחר).
- B מסניף את הפקטה בעזרת scapy.

2. יצירת טבלת ניתוב – כהגדרה סטטית או דינאמית:

טבלת הניתוב של נתב נוצרת כרשימה של מילונים בתוכנת הנתב.

כאשר משתמש רוצה להוסיף עמודה בטבלת ניתוב – בצורה סטטית:

- המשתמש מזין את כתובת ה subnet, IP, את המסכה הרלוונטית ואת ה Interface
- העמודה מוכנסת לטבלה ולפי גודל המסכה ממויינת הטבלה - לפי שיטת the longest match), ככל שהמסכה יותר גדולה היא תהיה יותר קרובה לראש הרשימה.

כאשר טבלת הניתוב מוגדרת במצב דינאמי:

- כל שלושים שניות (קבוע) כל הנתבים שולחים הודעת RIP לכל הנתבים המוכרים בנתב. הפקטה נבנית ב scapy, ומכילה את טבלת הניתוב של הנתב השולח.
- תוכנת הנתב מסניפה את ההודעות הללו ומחלצת מתוכם את הטבלאות ניתוב.

- כל פעם שהפקטה נקלטת בתוכנת הנתב, מתעדכנת טבלת הניתוב של הנתב, על פי זאת המתקבלת בפקטה, כך שכל ערכי ה metric של כל העמודות בטבלה יעלו ב 1.
- תכנת הנתב עוברת על העמודות של הטבלאות ואם מוצאת כי לכתובת subnet (כולל מסכה) מסוימת, ה metric מהטבלה שנשלחה יותר נמוך מזה הקיים בטבלה לאותה כתובת subnet, יחליף הנתב בטבלת ניתוב שלו את העמודה בה נמצאת כתובת ה subnet המדוברת, בעמודה עם אותה כתובת subnet הנמצאת בטבלת הניתוב שבתוך הפקטה.
- אם קיימת כתובת subnet בתוך הטבלה שנשלחה שאינה נמצאת כלל בטבלת הניתוב של הנתב המקבל, יכניס את העמודה של אותה כתובת לטבלה שלו בשינויים הנדרשים.

3. <u>ניתוב הפקטה – בעזרת הגדרת routing table סטטית או דינאמית:</u>

כאשר פקטה מגיעה לשלב בו היא צריכה להישלח מחוץ לרשת המקומית - לנתב:

- תכנת הנתב מחלצת (בעזרת scapy) את כתובת היעד של הפקטה
- תכנת הנתב עוברת על כתובת היעד על פי העמודות (כתובת ומסכה) הקיימות בטבלת ניתוב - מהראשונה לאחרונה - עבור כל כתובת בטבלה הנתב מבצע AND על כתובת היעד עם המסכה והתוצאה מושווה לכתובת המוצגת בטבלה.
- אם לאחר הפעולה התוצאה והכתובת שוות, הפקטה תישלח (בעזרת כימוס) ל Interface הרלוונטי המופיע בטבלה.
- אם אף אחד מהכתובות לא מתאימות, תשלח בחזרה הודעת שגיאה למשתמש, בה מפורט שלא הוגדר ניתוב לכתובת כזאת.

4. <u>הקמת ותחזוקת טבלאות ARP – על פי הסנפת פקטות ARP ב Scapy:</u>

טבלת ה ARP נשמרת בתוכנה רכיב הרשת, כמילון המכיל IP:MAC.

כאשר רכיב הרשת רוצה לשלוח לרכיב הרשת אחר/נתב הודעה ואין לו בטבלת ARP את הכתובת MAC שלו:

- רכיב הרשת בונה פקטת הפצה (broadcast) של בקשת ARP, עם scapy, המחפשת את כתובת ה MAC של רכיב הרשת האחר. הפקטה נשלחת לכל רכיבי הרשת ברשת הפרטית שלו.
- ההודעה מוסנפת על ידי רכיב המתג והוא כותב בטבלת SQL את נתינת ההרשאות להסנפת הפקטה כמפורט באלגוריתם 6.
- רכיבי הרשת קוראים את טבלת ההרשאות ובודקים אם יש להם הרשאה להסניף את הפקטה.
- רכיבי הרשת ברשת אשר מופיעים בטבלת ההרשאות הפרטית מסניפים את ההודעה, אם אין להם את כתובת ה MAC של רכיב הרשת השולח, מוסיפים לטבלת ARP שלהם (למילון) את הכתובת MAC המשויכת ל IP שממנו נשלחה הפקטה.

- רכיב הרשת שאליו נשלחה ההודעה עושה את הפעולה הקודמת ולאחר מכן בונה עם scapy פקטת ARP תשובה ובכתובת יעד של ה MAC מכניסה את הכתובת הפיסית שממנה נשלחה ההודעה.

5. <u>ניהול המתג – מילון וטבלת SQL:</u>

כדי ליישם מתג, השתמשתי במסד הנתונים SQL שבתוכו טבלת הרשאות הקובעת למי מבין המחוברים לרשת צריכה לעבור ההודעה. כשהמתג נוצר הוא מבצע את הפעולות הבאות:

- יוצר טבלת SQL של הרשאות.
- מתחיל תהליכון שקורא מקובץ טקסט אם המשתמש חיבר עוד מחשבים אליו.
- מסניף חבילות מסוג ARP שהכתובת יעד שלהם הוא אחד המחשבים המחוברים אליו בעזרת ספריית Scapy.
- בודק האם כתובת היעד מופיעה אצלו במילון המדמה את טבלת ה MAC של מתג. אם כן, מכניס לטבלת SQL של ההרשאות רק את רכיב היעד, אחרת מכניס לטבלה את כל הכתובות של המחשבים המחוברים ברשת הפרטית.

3.3. דרישות ואילוצי פתרון

לפרויקט מספר אילוצים:

- המערכת תרוץ רק על מערכות הפעלה מסוג Windows 7 או Windows 10 מכיוון שאלו שתי מערכות ההפעלה הממצות את כל היכולות של scapy הנדרשות בפרויקט.
- המערכת לא תתמוך בהגדרת יותר ממתג אחד לכל רשת פרטית.
- נדרשת התקנה מוקדמת של ספריית scapy בגרסה העדכנית ביותר שלה (2.4.5) מכיוון שהתקשורת בפרויקט גם של השרת וגם של הלקוחות מתבססת על ספרייה זו. קבלה של PORT פנוי ו IP לטובת תקשורת Socket מתבצעת בין השרת לשאר ה clients ברשת.
- בנוסף דרושה התקנה של גרסת python 3.x ו#C (WinForms) מכיוון שאלו השפות שבהן כתוב הקוד.
- המערכת לא תתמוך ב ARP דינאמי אלא תשלח הודעת ARP רק כאשר המשתמש מבקש לשלוח חבילת תקשורת, על פי דרישות הפרוטוקול.

לפרויקט מספר דרישות:

- הפרויקט נדרש לתמוך במינימום בפרוטוקולי תקשורת מסוג: RIP,NAT, TCP, ARP , ICMP,UDP.
- הפרויקט נדרש לתמוך בצורות טופולוגיות משתנות של רשת (סידור רכיבי רשת שונים וחיבורם) לפי הגדרת המשתמש בתנאי שניתן לממש תקשורת עם טופולוגיה זו, למשל הפרויקט לא יתמוך ברשת הבנויה רק ממתגים (ללא מחשבים כלל).

- הפרויקט נדרש לתמוך בביצועים במהירות מספקת למשתמש, כלומר ללא הבדלים (הנראים לעין) בין קיום התקשורת בין הרכיבים ובין הצגתה למשתמש.
- הפרויקט נדרש לתמוך בהסנפת התקשורת עם Wireshark – יהיה ניתן לעקוב אחרי התקשורת המתרחשת במערכת בעזרת הכלי Wireshark.

3.4. ממשקים למערכות חיצוניות

אין למערכת ממשקים למערכות חיצוניות.

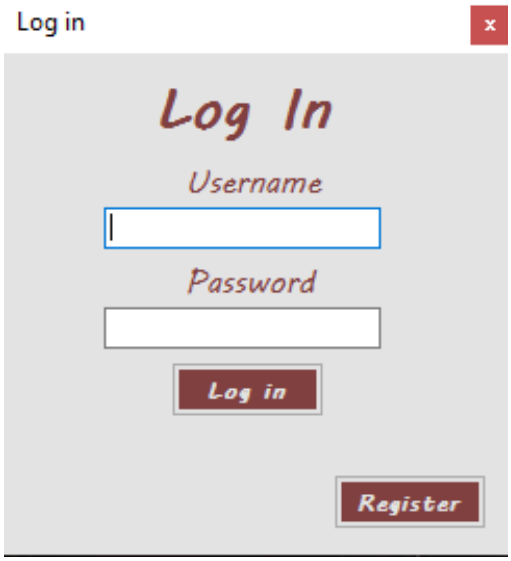
3.5. התייחסות לנושא אבטחה

בכניסה למערכת (ב ADMIN) יתבקש המשתמש להיכנס עם שם משתמש וסיסמא. כך, תגן המערכת מכניסת משתמשים לא מורשים (רשומים).

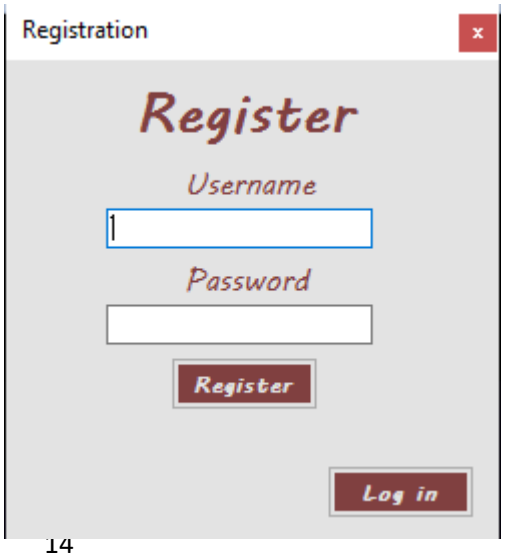
3.6. ממשק משתמש

1. חלונות כניסה:

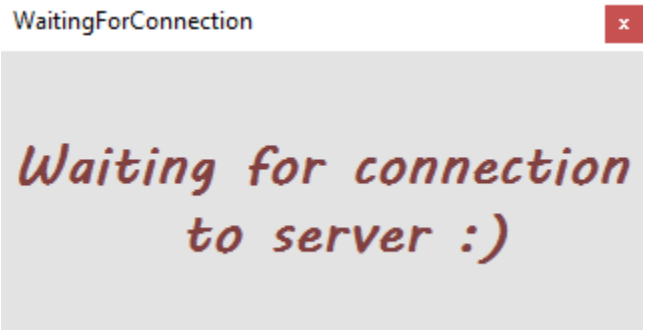
החלון הראשון שיופיע עם הרצת ה ADMIN, הוא חלון ההתחברות:



אם המשתמש אינו רשום, יצטרך המשתמש לעבור למסך ההרשמה בלחיצת על כפתור ההרשמה בצד הימני של מסך ההתחברות. להלן מסך ההרשמה:



לאחר הרשמת והתחברות המשתמש, יפתח מסך בו המשתמש יתבקש לחכות להתחברות עם השרת :



לאחר התחברות השרת יופיע החלון הראשי.

2. <u>חלון הראשי:</u>

החלון הראשי של אפליקציה מורכב מכמה חלקים. בתמונה זו ניתן לראות את החלון הראשי בשלמותו.

להלן פירוט של חלקי החלון הראשי:

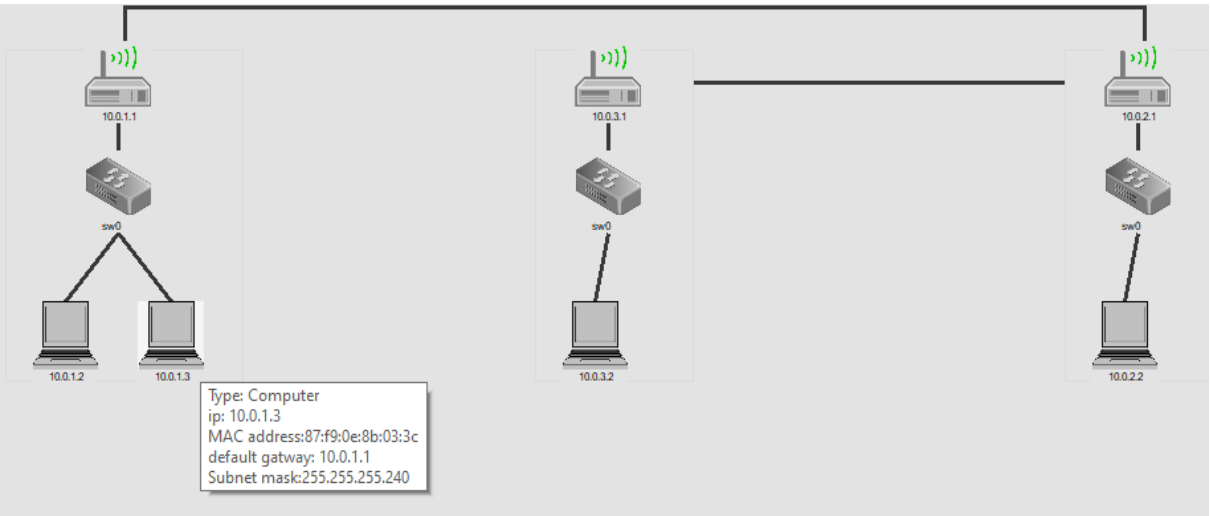
- בחלק השמאלי העליון ניתן לראות פירוט של הרשתות הפרטיות המחוברות במערכת כתצוגת עץ. הענפים המרכזיים בעץ הן הנתבים של הרשתות הפרטיות והן מכילות בתוכן ענפים המכילים מידע על הרשת – המחשבים המחוברים, המתג ברשת ולאיזה מחשבים מחובר והרשתות אליהן מחוברת

הרשת. חלק זה מתעדכן אוטומטית בכל פעם שהרשתות ברשת הווירטואלית מתעדכנות

- בחלק השמאלי התחתון ניתן להבחין בכפתורים של SAVE ו LOAD שתפקידם לשמור ולהעלות פרויקטים (הפרויקטים בפורמט שלי ושמורים כקובץ טקסט).

- החלק האמצעי העליון של המסך הוא "מגרש המשחקים" של המשתמש. בו ניתן לראות את הרשתות הפרטיות השונות ברשת הווירטואלית, את החיבורים בניהן ובין הרכיבים בתוך הרשת הפרטית ואת התהליך אותו עוברת התקשורת ברשת.

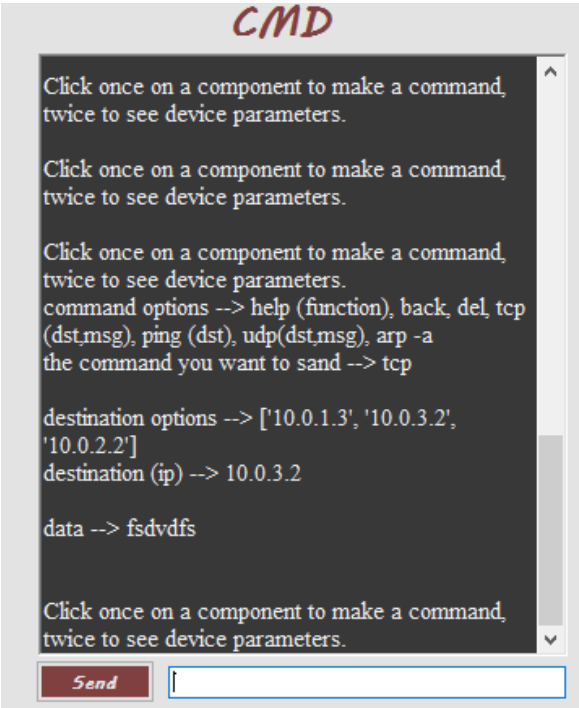
להלן דוגמה לשלוש רשתות מחוברות המורכבות ממחשבים, נתבים ומתגים:



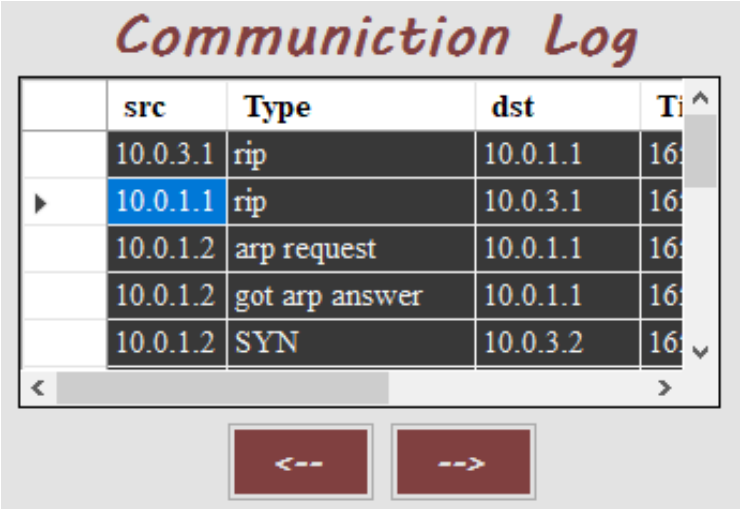
כאשר המשתמש "מרחף" עם מצביע העכבר מעל הרכיב, מוצגים פרטים כללים על הרכיב. ניתן ללחוץ על האיור של הרכיב פעם אחת כדי לבצע בו פקודות (בשורת פקודה) ופעמיים כדי לפתוח חלון עם פרטים מורכבים על הרכיב.

- בחלק האמצעי התחתון של המסך ניתן לראות את המקרא שמסביר למשתמש את המשמעות של כל איור של רכיב:

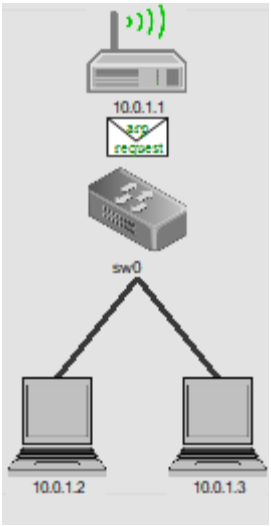
- בחלק הימני העליון של המסך ניתן לראות את "שורת הפקודה", בה המשתמש (לאחר לחיצה על אחד הרכיבים) רושם הוראות למערכת:



- בחלק הימיני התחתון של המסך ניתן לראות את תיעוד התעבורה ברשת הוירטואלית המיוצג כטבלה. כל שורה בטבלה מתארת חבילה שעברה ברשת.



לטבלה 5 עמודות – כתובת המקור, סוג החבילה, כתובת היעד, זמן השליחה/קבלה והודעה אם החבילה מכילה כזאת.
על ידי לחיצה על הכפתורים בתחתית המסך, המשתמש יכול לראות את התהליך שאותה עוברת כל חבילה. בכל לחיצה על אחד הכפתורים (קדימה בשביל הצגת החבילה הבאה ואחורה הצגת בשביל החבילה הקודמת), תודגש שורה בטבלת ה LOG ותוצג אנימציה של מעטפה העוברת בין רכיבי ברשת בהתאם למסלול אותה עברה החבילה אותה מייצגת השורה המודגשת בטבלה.

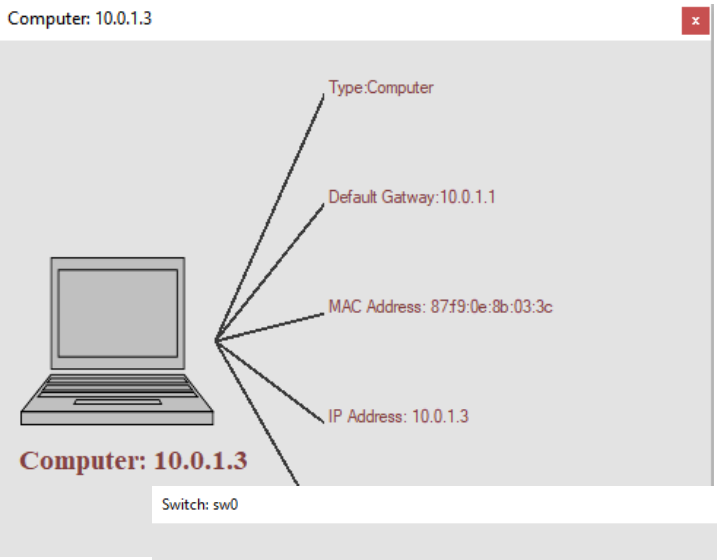


דוגמה לחבילה העוברת ברשת ומיוצגת בצורה גרפית כמעטפה (אנימציה):

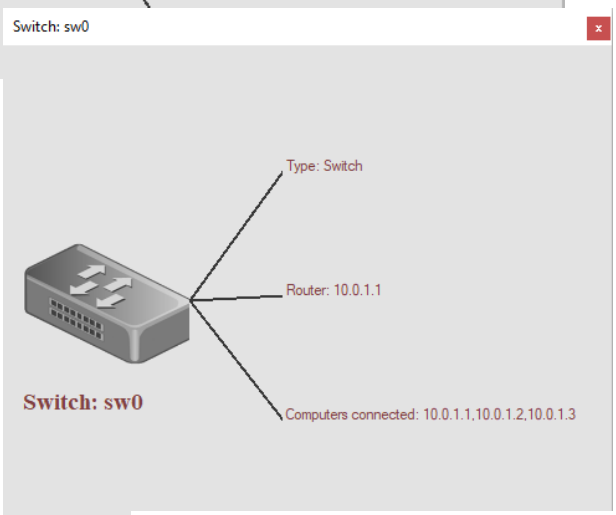
3. חלון משני – הצגת רכיב:

כאשר לוחצים פעמיים על איור של רכיב ברשת הפרטית נפתח חלון עם פרטים מורכבים על הרכיב:

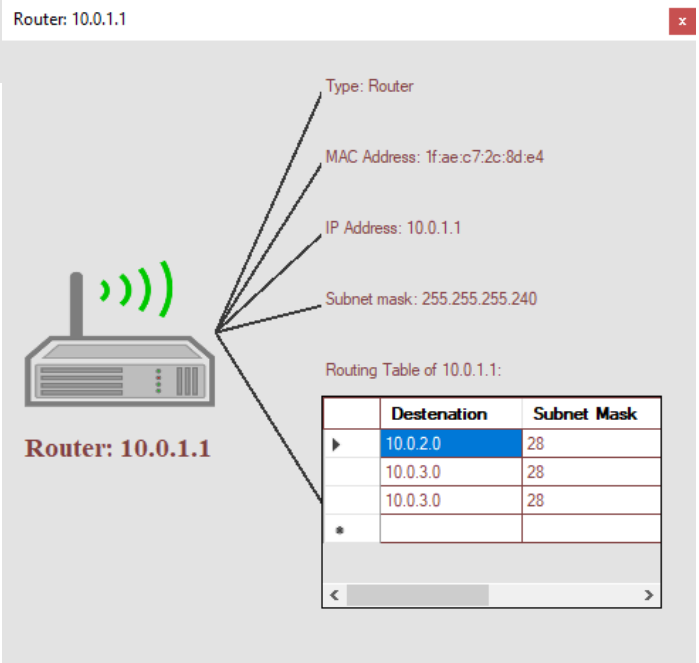
תצוגה של מחשב:



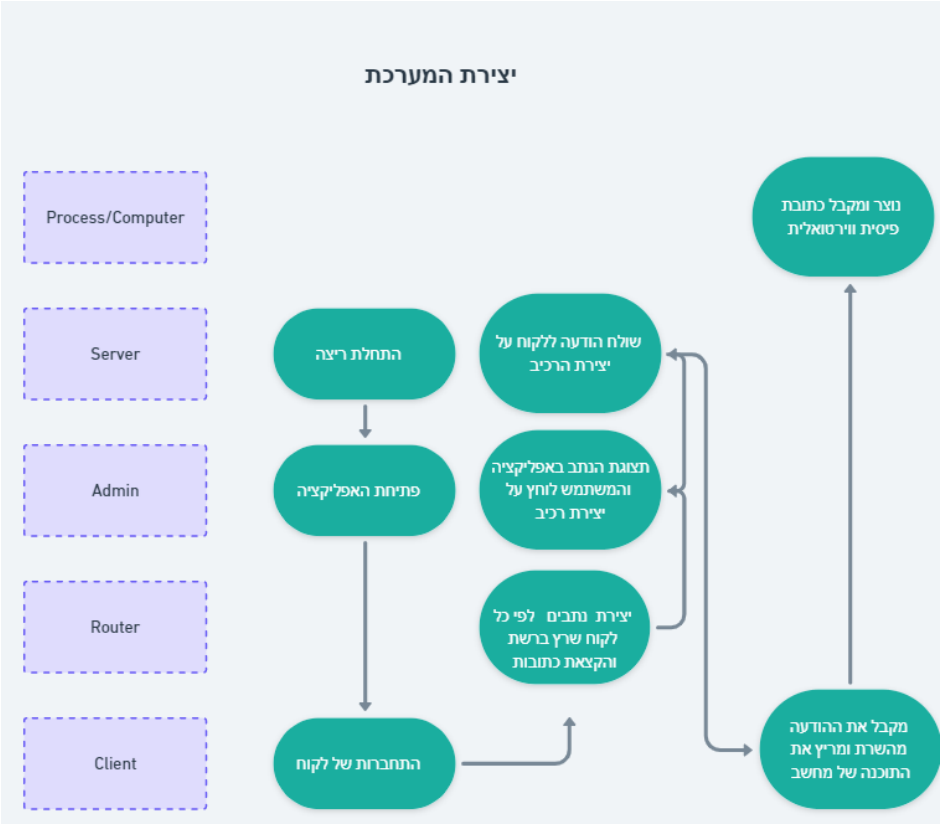
תצוגה של מתג:

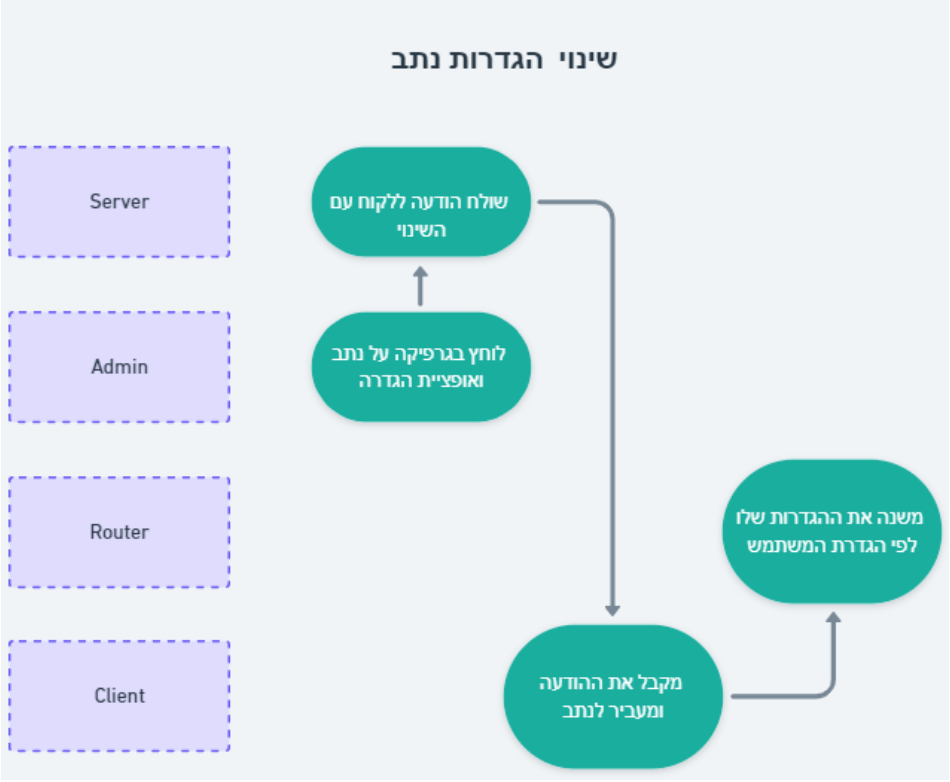


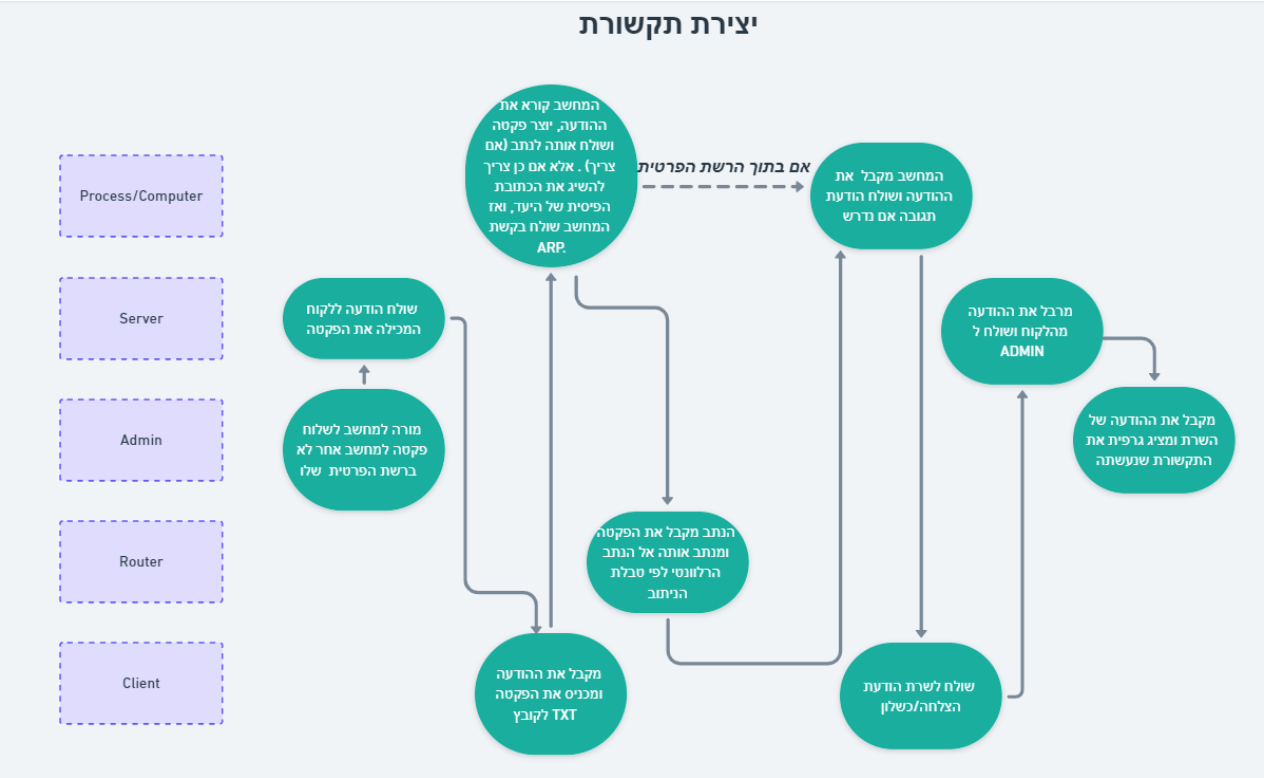
תצוגה של נתב:



3.7. תרחישים עיקריים







4. תהליך כתיבת הפרויקט

4.1. תהליך הפרויקט

התחלתי את הפרויקט מיד כשניתן בתחילת השנה. ידעתי שאני רוצה שהפרויקט שלי יעסוק ברשתות אך התלבטתי האם אני רוצה להתמקד בפרויקט בתחום האבטחה או לא. לאחר מחשבות והתייעצות רבה עם המורה המנחה, החלטתי לעשות פרויקט - Packet Tracer רשתי ללימוד רשתות וסייבר. בהתחלה כהוכחה שאני מסוגל לבצע את הפרויקט (POC), יצרתי תצורת מערכת של שתי רשתות פרטיות שבכל אחת נתב ומחשב. המטרה שלי הייתה ליצור תקשורת בין המחשבים ברשתות השונות. צלחתי מסוכה זו.

לאחר מכן יצרתי מערכת יותר מורכבת עם שרת שינהל את המערכת, ולקוחות שינהלו כל אחד את הרשת הפרטית שרצה במחשב נפרד. בשלב הבא יצרתי גם את רכיב המתג, שמימושו במערכת דרש מחשבה מעמיקה מכיוון שמתג פועל ברמה 2 במודל ה OSI, רמה שרשת ווירטואלית כמו שאני יצרתי מתקשה לתמוך בה בצורה שהיא מיושמת במציאות. לאחר מחשבה ומחקר מקיף, יישמתי את הרכיב כטבלת הרשאות עם מסד הנתונים SQL, אשר מנוהלת על ידי מודול ה Switch שבניתי.
בשלב הבא בפרויקט עסקתי ביישום פרוטוקולים שונים כמו RIP, ICMP,TCP,UDP, ו-ARP כדי שרכיבי המערכת יתפקדו כמו רכיבי רשת אמיתיים. בשלב זה ייצרתי גם ממשק משתמש זמני לנוחותי האישית. לבסוף נותר לי ליישם סטנדרט גבוה ועיצוב מרשים של ממשק המשתמש, שגם הווה אתגר לא פשוט, לאור ניסיוני המועט בכלי WinForms.

בנוסף ללמידה מעמיקה ומשמעותית של החומר שהפרויקט עוסק בו, למדתי המון על כתיבת פרויקט. יש שוני רב בין כתיבת תוכנה קטנה, לבין כתיבת מערכת גדולה ומורכבת. אני חש כי כתיבת עבודה זו הקנתה לי ידע רב ומיומנות שימושית לעתידי.

4.2. אתגרים ואופציות שונות למימוש

במהלך כתיבת הפרויקט נתקלתי בכמה אתגרים:

- בהתחלה היה לי אתגר בבחירה האם לממש את הרשת הווירטואלית על מחשב אחד כמו Cisco Packet tracer ואז ליצור תקשורת מדומה בשילוב ניהול מבני נתונים, או לממש את המערכת על כמה מחשבים שכל אחד ייצג רשת פרטית המקושרת למחשב עליו יורץ הלקוח. הגעתי להבנה שאני רוצה שהמערכת תשמש ככלי ללימוד רשתות ולא ככלי שבעזרתו מתכננים מערכות מורכבות – שכך, מערכת המורכבת מכמה מחשבים שונים, למרות כל מגבלותיה, התאימה לרעיון (ואף אפשרה צפייה ב Wireshark). שיקול נוסף התייחס לרצון שלי לחדש וליצור משהו חדש משלי ולא להתחרות במוצר קיים.

- במהלך כתיבת הפרויקט נתקלתי בבאג של ספריית Scapy, שבגללו לא יכולתי להתקדם שבועיים. הבעיה הייתה שלאחר הסנפת חבילה הייתי צריך לבצע סדרה של פעולות, מה שגרם לפספוס חבילות בזמן ביצוע הפעולות הללו. הצלחתי לפתור זאת באמצעות prn – פונקציה מובנת של הספרייה הנותנת להריץ בהליכון נפרד פונקציה המקבלת את החבילה כפרמטר. בנוסף הייתי צריך לעשות סינון ספציפי לאיזה חבילה אני רוצה להסניף. כך, לאחר הרבה תסכול וניסוי וטעיה הצלחתי לפתור את בעיה זו.

- כפי שציינתי לעיל, מימוש רכיב המתג ברשת ווירטואלית היווה אתגר לא פשוט. מאחר שהוא מעביר את התקשורת בעזרת port פיזי, לא הייתה אפשרות לממש מתג

כפי שהוא ממומש במציאות. לכן הייתי צריך למצוא דרך יצירתית לממש את המתג כאשר מצד אחד אני רוצה לפתור את הבעיה בצורה תוכנתית עקיפה ומצד שני לא לחטוא לדרך האמיתית בה עוברת התקשורת ברשת הפרטית. עמדה בפני האפשרות לתמוך במתג רק בצד של המשתמש, כלומר רק בגרפיקה, ובפועל להשתמש במבני נתונים של המחשבים במערכת כדי לדמות ניהול של מתג מאחורי הקלעים. לא בחרתי בפתרון זה מכיוון שהרגשתי שזה לא נאמן לתיאוריה ומכיוון שאחת המטרות המרכזיות בפרויקט היא לבצע את התקשורת כפי שהיא נעשית באמת ברשת פרטית, לבסוף, בחרתי ליישם את המתג בתור מסד נתונים SQL שינהל טבלת הרשאות אשר בתוכה יוצגו אילו רכיבים מורשים להסניף כל חבילה – דבר המדמה הפצה של חבילות דרך ports שונים כפי שנעשה במציאות.

- התלבטתי אם לממש את הגרפיקה של הפרויקט ב pyqt או ב winforms . רציתי לממש את הגרפיקה ב pyqt כדי להיצמד לשפת תכנות אחת. בנוסף, מעולם לא התנסיתי ב winforms. לבסוף החלטתי לממש את הגרפיקה ב winforms לאור העובדה שהכלי נוח יותר לשימוש, והוא בעל אפשרות עריכה ב designer, ואפשרות צפיה ותכנון של מיקום הרכיבים השונים ללא הרצה.

5.מרכיבי פתרון

5.1. תיחום הפרויקט

- תקשורת – שרת, לקוח, יצירה והסנפת תקשורת בעזרת Scapy של פרוטוקולים שונים ברמות 2 ו3 של מודל ה OSI, ניתוחה של התקשורת ויישום רכיבי תקשורת – נתב, מתג, מחשב (כולל טבלאות ניתוב ו ARP).
- אבטחת מידע – כניסה מאובטחת של משתמש לאפליקציה בעזרת שם משתמש וסיסמה.
- תצוגה – גרפיקה מרשימה וידידותית למשתמש שנכתבת בשפת #C (WinForms).
- מבנה נתונים – שמירה ברשימות, מילונים ומשתנים גלובליים כדי לשמור את המידע על מחשבים ברשת וכדי לשמור את המידע הדרוש כדי לקיים את התעבורה ברשת.
- מערכות הפעלה – שימוש רב ב Threads ו Processes
- ארכיטקטורת קוד – שימוש רב במחלקות, קבצים ומימוש של מודל ה MVC. כל זאת כדי להקל על תהליך כתיבת הקוד וגמישות בשינויים.
- תיעוד – תיעוד אקטיבי וניהול GIT מוקפד (שמירת גרסאות שונות), בנוסף לכתיבת ספר פרויקט.

5.2. סביבת העבודה (טכנולוגיה)

שפות התכנות:

קוד השרת והלקוחות ייכתב בשפת Python 3.x זאת מכיוון ש Scapy נתמכת רק ב Python וזאת ספרייה שעלייה מבוססת כל התקשורת בין הרכיבים השונים בפרויקט.

תוכנת מנהל הרשת, ה ADMIN, שעלייה רץ ה GUI כתובה בשפת #C ומשתמשת באופציית ה GUI הנוחה ש Visual Studio מאפשר – WinForms. החלטתי להשתמש דווקא בגרפיקה ב #C WinForms בגלל פשטות הכתיבה ש #C מאפשרת, האופציות המרובות הנתמכות בתוכנה והרמה הגרפית הגבוהה של WinForms, שכן הגרפיקה מהווה חלק נכבד בפרויקט.

בנוסף, על מנת לבדוק ולבדוק את התוכנה בזמן כתיבת הפרויקט איעזר ב Wireshark - תוכנת "רחרחן", על מנת לצפות בפעילות התוכנה ברשת. בנוסף אמליץ למשתמש המריץ את הפרויקט להריץ אותו יחד עם Wireshark וכך יוכל לראות את הפקטות האמיתיות שהוא הגדיר ושלח במו ידיו.

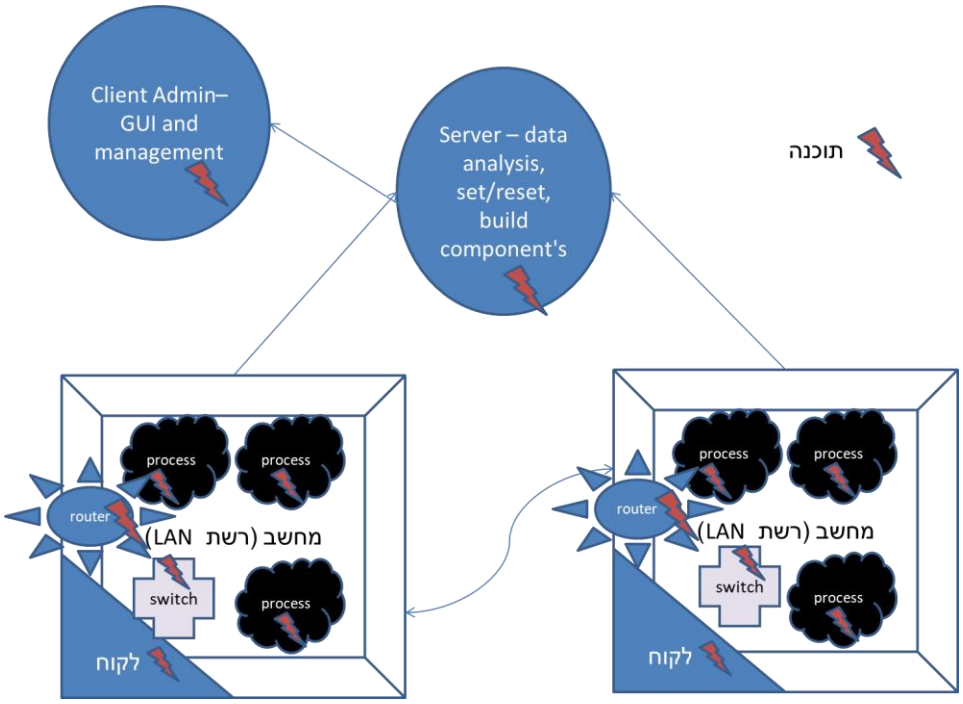
סביבות פיתוח:

Visual Studio לשפת #C

Pycharm לשפת Python

5.3. מבט טופולוגי

המערכת מורכבת משרת, מ ADMIN המציג את הגרפיקה ומכיל תוכנות הרצות כאשר מתחבר לקוח למערכת. שתי המסגרות מייצגות שתי מערכות (רשתות פרטיות), שבהם מורץ לקוח ותוכנות של רשת פרטי, המורכבת מנתב, מתג ושלושה מחשבים. העיגולים מיצגים את השרת וה ADMIN. השרת מחובר לשני הלקוחות וה ADMIN מעל תקשורת SOCKET.



5.4. מבנה נתונים

Server:

- computers_connected - מילון שבו נשמרים כל המחשבים המחוברים לשרת.
- sub_comp - מילון בו נשמרים כל המחשבים הווירטואליים השייכים לרשת המורצת מכתובת IP של לקוח מסוים.
- switchs – מילון של כל המחשבים המחוברים למתג השייך לרשת מכתובת IP של לקוח מסוים.
- lans – מילון של הנתבים עם כתובות הרשת שלהם והמסכות.
- sub_comp_out – מילון השומר את כל המחשבים הווירטואליים של לקוח שיצא/ נותק (שומר מצב).
- routing_tables – מילון עם טבלאות הניתוב של כל נתב וירטואלי.
- arp_tables – מילון עם כל טבלאות ה ARP של כל מחשב ברשת הווירטואלית.
- ip_mac – מילון המשייך כתובת IP לכתובת MAC של רכיב וירטואלי ברשת.

Client:

- computers – מילון המכיל את הכתובות IP וכתובות MAC לכל רכיב ברשת הווירטואלית.
- process_manager – מילון השומר בתוכו את כל ההליכונים של רכיבי הרשת מול כתובות ה IP שלהם.

Router:

- router_table – מילון המהווה טבלת ניתוב של הנתב.
- arp_table – מילון המהווה טבלת ARP של הנתב.

Switch:

- connected – רשימה של הרכיבים המחוברים למתג.
- connections – מילון של כתובות IP של רכיבים מול כתובות MAC שלהם המהווה כטבלת ה MAC של המתג.

Process:

- arp_table – מילון המהווה טבלת ARP של רכיב הרשת.

Admin:

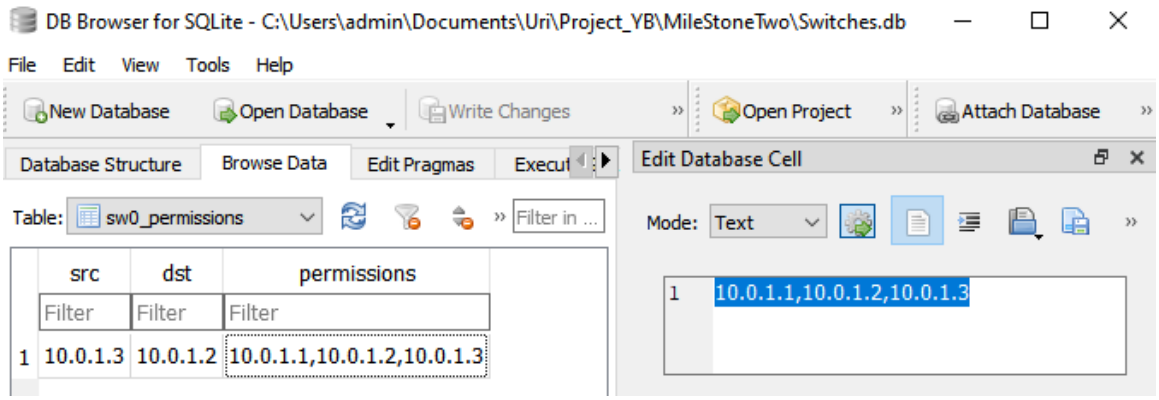
- lans – מילון השומר את אובייקט ה GroupBox הרלוונטי לכל רשת פרטית במערכת.

- counters – מילון השומר מונה של כל רשת פרטית במערכת המונה את מספר המחשבים ברשת.
- compConnected – מילון של הרכיבים המחוברים למתג בכל רשת פרטית.
- routingTabels – מילון השומר את טבלאות הניתוב של כל נתב ברשת.
- macTabels – מילון השומר את הכתובות MAC של הרכיבים ברשת.
- comQueue – תור של התקשורת הרצה ברשת. מבנה נתונים האחראי על שמירת התקשורת שמקבל המשתמש ולתרגם אותה לאנימציה של החבילה העוברת ברשת.

5.5. מסד נתונים

קיים מסד נתונים מסוג SQL המהווה יישום של המתג במערכת. הוא מנוהל על ידי מודל sqlite3. במסד זה נשמרים נתונים בטבלה:

- טבלת sw0_permissions השומרת את נקודות הקצה המורשות לקלוט כל חבילה. כל שורה בטבלה מייצגת חבילה שנשלחה. העמודות בטבלה הן src המייצג את כתובת השולח, dst המייצג את כתובת היעד של השולח ו permissions המייצג את הכתובות המורשות לקלוט את הפקטות.
-



5.6. מבט מודולרי

החלקים המרכזיים בפרויקט:

- Server – השרת אשר שומר במבני נתונים את כל המידע שמועבר במערכת ומקשר בעזרת sockets בין כל הלקוחות וה ADMIN (שהוא המשתמש). ה Server מורץ כקובץ פייתון.
- Client – מדווח על הנעשה ברשת פרטית (ווירטואלית) במחשב עליו מורץ לשרת בעזרת socket המחובר לשרת. מריץ מהליכונים מתוכו את הרכיבים השונים ברשת. כאשר מורץ נוצרת רשת חדשה אשר אוטומטית (בהנחה שה Server וה ADMIN מורצים) מוצגת אצל המשתמש. ה Client מורץ כקובץ פייתון.
- ADMIN – תצוגת המערכת והרשת הווירטואלית למשתמש בצורה גרפית. דרך התצוגה המשתמש יכול לתת פקודות ולשנות את המערכת כרצונו. ה ADMIN

מחובר אל ה Server בעזרת socket. ה ADMIN מורץ בתור קובץ C# Winforms.

- Router – הרכיב המייצג את נתב במערכת. מנתב את התקשרות מחוץ לרשת הפרטית. מורץ אוטומטית כהליכון בתוך Client. קובץ פייתון.
- Switch - הרכיב המייצג מתג במערכת. מורץ כאשר המשתמש מגדיר למערכת מתג. מורץ כהליכון בתוך Client. קובץ פייתון.
- Process - הרכיב המייצג מחשב במערכת. מורץ כאשר המשתמש מגדיר למערכת מתג. מורץ כהליכון בתוך Client. קובץ פייתון.

5.7. פירוט מודלים עיקריים

Class	Function	Input\Output	Description
server	ip_generator	input: string output: string	יוצר כתובת IP חדשה לרשת פרטית.
	mac_generator	input: None output: string	יוצר כתובות MAC חדשות.
	update_admin	input: socket, string output: None	מעדכן את ה ADMIN בשינויים שנעשו במערכת.
	new_command	input: string, socket, string output: string	יוצר פקודה ל Client על יצירת רכיב רשת חדש (מחשב או מתג).
	delete_command	input: string, socket, string output: string	יוצר פקודה ל Client על מחיקת מחשב מהרשת הפרטית אליה הוא שייך.
	delete_switch_command	input: string, socket, string output: string	יוצר פקודה ל Client על מחיקת מתג מהרשת הפרטית אליה הוא שייך.
	connect_switch_command	input: string, socket, string output: string	יוצר פקודה ל Client על חיבור רכיב תקשורת למתג מהרשת הפרטית אליה הוא שייך.
	rout_add_command	input: string, socket, string output: string	יוצר פקודה ל Client על חיבור של שני רשתות --הוספת שורה לטבלת ניתוב של נתב.
	showrp_command	input: string, socket, string output: string	יוצר פקודה ל ADMIN שמחזירה טבלת ARP של רכיב ברשת.
	tcp_command	input: string,	יוצר פקודה ל Client על יצירת תקשורת

		socket, string output: string	TCP בין שני רכיבים ברשת.
	udp_command	input: string, socket, string output: string	יוצר פקודה ל Client על יצירת תקשורת UDP בין שני רכיבים ברשת.
	ping_command	input: string, socket, string output: string	יוצר פקודה ל Client על יצירת תקשורת PING בין שני רכיבים ברשת.
	dell_all	input: None output: None	פעולה המאתחלת את המערכת – מאתחלת את מבני הנתונים בשרת ושולחת לכל הרכיבים ברשת להימחק.
	update_comp	input: string, socket output: None	מעדכן רשת פרטית אחת ספציפית שהלקוח שלה נותק על המצב שלה לפני הכיבוי.
	update_struct	input: string, socket output: None	מעדכן את כל המערכת על שינוי שנעשה ברשת.
	input_user	input: None output: None	פונקציה שרצה כל עוד המערכת פועלת. מטרתה לקבל פקודות מהמשתמש (ה ADMIN) ולשלוח ללקוח הרלוונטי את הפקודה הרלוונטית.
	sock	input: None output: None	פונקציה שרצה כל עוד המערכת פועלת. מטרתה לקבל מידע מהלקוחות (על תקשורת, חיבור, התנתקות וכדומה...)
	analysing_commun ication	input: string, string output: None	פונקציה המנתחת את התקשורת ברשת ומעדכנת את מבני הנתונים של השרת בהתאם.
Client	set_up	input: None output: None	יוצר קובץ מסד נתונים וקבצי TXT רלוונטיים למערכת.
	check_if_in_lan	input: string output: bool	בודק אם כתובת IP מסויימת שייכת לרשת הפרטית הוירטואלית במחשב הלקוח.
	update_switch	input: string output: None	מעדכן את רכיב המתג (בעזרת כתיבה לקובץ TXT)
	create_switch	input: string output: None	יוצר מתג לרשת הפרטית הווירטואלית.
	create_new_proces	input: string	יוצר מחשב לרשת הפרטית הווירטואלית.

	s	output: None	
	create_router	input: None output: None	יוצר נתב לרשת הפרטית הווירטואלית.
	client_listener	input: None output: None	פונקציה הפועלת כל עוד הלקוח רץ. הפונקציה קוראת את הקבצי TXT איתם מתקשר עם הרכיבי תקשורת במערכת.
	delete_process	input: string output: None	מורה על מחשב וירטואלי במערכת להימחק.
	delete_switch	input: string output: None	מורה על מתג וירטואלי במערכת להימחק.
	send_tcp	input: string output: None	מורה על יצירת תקשורת מסוג TCP.
	send_ping	input: string output: None	מורה על יצירת תקשורת מסוג PING.
	send_udp	input: string output: None	מורה על יצירת תקשורת מסוג UDP.
	receive	input: None output: None	פונקציה הפועלת כל עוד הלקוח רץ. מקבלת הודעות מהשרת ומבצעת את הפונקציה הרלוונטית.
Router	check_if_in_lan	input: string output: bool	בודק אם כתובת IP מסויימת שייכת לרשת הפרטית הוירטואלית של הנתב.
	check_with_switch	input: string output: bool	בדיקה עם מסד הנתונים של המתג האם יש הרשאה לקלוט פקטה מסויימת.
	update_arp_table	input: None output: None	רץ תמיד כשהנתב קיים. מקולט פקטות ARP ומעדכן את טבלת ה ARP של הנתב.
	arp_request	input: string output: string	שולח בקשת ARP למחשב מסויים ומחכה לקבלה. מחזיר את כתובת ה MAC של המחשב לאחר התגובה.
	send_rip	input: None output: None	שליחת RIP כל 30 שניות ומימוש הפרוטוקול. ניתוב דינאמי.
	sniff_rip	input: None output: None	קליטת חבילות RIP הנשלחות לכתובתו החיצונית של הנתב ומימוש הפרוטוקול. ניתוב דינאמי.
	update_client_communication	input: string, string output: None	מעדכן את הלקוח בתקשורת שנעשתה במערכת (עם הוספת חותמת זמן).

	listener	input: None output: None	פונקציה הפועלת כל עוד תוכנת הנתב רצה. הפונקציה קוראת את הקבצי TXT איתם מתקשר עם הלקוח המקושר לרשת הפרטית של הנתב.
	check_rout	input: string output: bool	בודק אם כתובת מסויימת קיימת בטבלת ניתוב של הנתב (האם אפשר לנתב חבילה לכתובת יעד מסויימת).
	rout_tcp	input: string, string output: None	ניתוב חבילות TCP למחשבים מחוץ לרשת הפרטית.
	rout_icmp	input: string, string output: None	ניתוב חבילות ICMP למחשבים מחוץ לרשת הפרטית.
	rout_udp	input: string, string output: None	ניתוב חבילות UDP למחשבים מחוץ לרשת הפרטית.
	sniff_packet_outlan	input: None output: None	קליטת חבילות הנשלחות לרשת הפרטית מרשתות אחרות.
Switch	create_table	input: None output: None	יצירת טבלת הרשאות בתוך מסד הנתונים (SQL).
	sniffer	input: None output: None	בודק את החבילות הנשלחות ברשת ומעדכן את טבלת ההרשאות במסד הנתונים.
	update_connections	input: None output: None	מעדכן חיבורים חדשים של מחשבים וירטואלים למתג.
Process	check_if_in_lan	input: string output: bool	בודק אם כתובת IP מסויימת שייכת לרשת הפרטית הוירטואלית של הרכיב תקשורת.
	check_with_switch	input: string output: bool	בדיקה עם מסד הנתונים של המתג האם יש הרשאה לקלוט פקטה מסויימת.
	update_arp_table	input: None output: None	רץ תמיד כשהנתב קיים. מקולט פקטות ARP ומעדכן את טבלת ה ARP של רכיב התקשורת.
	arp_request	input: string output: string	שולח בקשת ARP למחשב מסויים ומחכה לקבלה. מחזיר את כתובת ה MAC של המחשב לאחר התגובה.
	send_recieve	input: None	פונקציה הפועלת כל עוד תוכנת הנתב

		output: None	רצה. הפונקציה קוראת את הקבצי TXT איתם מתקשר עם הלקוח המקושר לרשת הפרטית של רכיב התקשורת.
	sniff_packet	input: None output: None	קליטת חבילות הנשלחות לכתובת של רכיב התקשורת.
	update_client_communication	input: string, string output: None	מעדכן את הלקוח בתקשורת שנעשתה במערכת (עם הוספת חותמת זמן).
	send_TCP	input: string, string output: None	יצירת תקשורת TCP.
	send_PING	input: string, string output: None	יצירת תקשורת PING.
	send_UDP	input: string, string, output: None	יצירת תקשורת UDP.
Admin	AddComponent	input: string, string, PictureBox output: null	הוספת איור של רכיב על המסך (במקום הנכון).
	CompClick	input: object, EventArgs output: null	מאורע כאשר המשתמש לוחץ על איור של רכיב – נשלחת הודעה ללקוח כי זה הרכיב שנבחר.
	CompDoubleClick	input: object, EventArgs output: null	מאורע כאשר המשתמש לוחץ לחיצה כפולה על איור של רכיב – נפתח חלון חדש עם פרטים על רכיב התקשורת.
	AnalyseCommunication	input: string output: null	מקבל תקשורת ומוסיף אותה לתור של התקשורת. מעדכן את המבני נתונים הרלוונטים.
	ChooseReceive	input: string output: null	מקבל את ההודעות מהשרת מסווגת אותם – לדיווח שתקשורת או עדכון של הרשת. מעדכן את המבני נתונים ואת הגרפיקה בהתאם.
	SendServerUpdate	input: string,string output: null	שולח לשרת עדכונים על שינויים שהמשתמש עשה במערכת.

מקבל קובץ בפורמט של המערכת ומעדכן את המערכת ואת השרת בהתאם.	input: string output: null	ImportFile	
מאורע הקורה כאשר רכיב מתחבר למתג – מצייר קו ישר העובר בין המתג לרכיב.	input: object, PaintEventArgs output: null	Lan_Paint	

6. תסריטי בדיקה

6.1. דגשים בבדיקה

- חיבור וניתוק של לקוחות מהמערכת ללא באגים – הצגה גרפית אם רשת התנתקה.
- טיפול במספר לקוחות.
- מעבר של הודעות בין ה Admin הלקוחות השרת והרכיבי רשת ללא באגים או בעיות אחרות.
- המערכת מבצעת את פקודות המשתמש.
- ביצוע תקשורת שניתן לצפות בה בעזרת הכלי WireShark.
- דיווח נכון של התקשורת הנעשית במערכת (מבלי לפספס חבילות).
- הצגת פרטי הרכיבים ברשת בצורה נכונה.
- המערכת מייבאת ושומרת קבצים.

6.2. תסריטי בדיקה עיקריים

- חיבור 2 לקוחות (2 רשתות) והגדרה ידנית של 2 מחשבים ומתג המחובר למחשב ונתב בכל מערכת (המתג מחובר לכל הרכיבים במערכת פרט למחשב אחד).
- חיבור של הנתבים באופן ידני בעזרת גדרת ניתוב סטטי לשניהם.
- שליחת הודעת Ping למחשב הלא מחובר וקבלת הודעת שגיאה Icmp unreachable.
- חיבור המחשב, הצלחת התקשורת והצגת התהליך באנימציה (הרצה קדימה ואחורה).
- ייבוא של פרוייקט מוכן מראש. איפוס של המערכת.
- ניתוק וחיבור כבל אינטרנט של מחשב של אחד הלקוחות, שיקוף גרפי של המאורע.
- חיבור של עוד לקוח (רשת) וחיבור ידני (סטטי) רק לאחד מן הנתבים הקיימים.
- שליחת הודעת TCP ממחשב ברשת החדשה למחשב ברשת שלא חובר אליה. קבלת הודעת שגיאה Icmp network unreachable.
- קבלת הודעת RIP וביצוע חוזר של התקשורת כולל שיקוף ב LOG של כל התהליך.

7. רפלקציה

7.1. לוח זמנים מוערך לניהול הפרויקט:

נובמבר	POC הכולל אפשרות העברת הודעת TCP בין 2 נקודות קצה שלא נמצאים באותה רשת פרטית, הגדרת טבלת ניתוב סטטית, וממשק שורת פקודה הכולל הצגת מידע על הרשת.
דצמבר	בניית Switch, יישום פרוטוקול ARP, מבנים שונים של רשת (חיבור נתבים בתצורה שונה), יישום תקשורת בין נקודות קצה בתוך הרשת הפרטית.
ינואר	יישום טבלת ניתוב סטטית ודינאמית. בנוסף, הקמה וסיום קשר של הלוקחות מול השרת בצורה נקייה (ללא שגיאות).
פברואר	בניית ניתוח של הרשת, אפשרות הגדרה ושחזור של הרשת. הגדרת פרוטוקולים מרכזיים – PING, UDP ו TCP.
מרץ	גרפיקה בסיסית וטיפול במקרי קצה (ניהול שגיאות)
אפריל	ניתור התקפות סייבר, וגרפיקה משופרת
מאי	גרפיקה מרשימה עם כל התרחישים, מוצר מוגמר (וספר מוגמר).

7.2. אתגרים ותרומה אישית

הפרויקט הוא ללא ספק התוצר הכי גדול שעשיתי בחיים שלי. בתחילת השנה שהגשתי את הצעת הפרויקט היה לי קשה להאמין שאצליח לעשות את כל מה שרציתי לעשות בעיקר כשידעתי שהשנה הולכות להיות לי מחויבויות רבות, ולכן גם לחץ זמן. כתיבת מסמך התכנון (PRD) וקביעת לוח זמנים נתנה לי תקווה שאני כן יכול להשלים את הפרויקט. החלטתי לעבוד מסודר, ולעשות לכל אבן דרך רשימת "דברים שצריך לבצע". בעזרת שיטה זו הצלחתי להגיע לפרויקט שאני מסופק ממנו.

כתיבת הפרויקט לימדה אותי המון. כשהתחלתי את הפרויקט אחד האתגרים המרכזיים שעמדתי בפניהם הייתה ללמוד לבד חומר שלא ידעתי וליישם אותו בלחץ זמן על הצד הטוב ביותר. אני מרגיש שעמדתי באתגר ודווקא הלחץ הוא זה שגרם לי להתעלות ולהצליח ללמוד לנהל את הזמן שלי יותר טוב.

הפרויקט הוא דבר שהעסיק אותי ביום יום. הייתי מוצא את עצמי בטיול או בשיעור אחר חושב על הפרויקט וכותב לעצמי הערות ורעיונות שהיו לי לשיפור הפרויקט. הייתה לי הרבה מוטיבציה ללמוד, לפתח ולהצליח בפרויקט גם בגלל אהבתי לנושא בו הפרויקט עוסק, וגם בגלל הרצון שלי לעמוד במטרות ובלוח זמנים שהגדרתי לעצמי.

7.3. תובנות

כתיבת הפרויקט הייתה קשה ומפרכת. בתחילת התהליך הייתי חדור מוטיבציה, תקווה ואמונה שאצליח לעשות את מה שאני רוצה לעשות. ככל שהתקדם התהליך כך הבנתי את כמות העבודה וההשקעה שהעבודה הזאת דורשת. הבנה זו גרמה לתסכול וייאוש וברגעים כאלה המוטיבציה לעמוד בלוח זמנים שקבעתי לעצמי היא מה שהניע אותי. עכשיו, אחרי שסיימתי את העבודה אני שמח שלא התייאשתי, אני רואה כמה למדתי והתפתחתי וכמה אני כל כך יודע עכשיו יותר ממה שידעתי כשהתחלתי את הפרויקט. תובנה זו גורמת לי לרצות לעבור את התהליך שוב ולהעריך אותו הפעם יותר. אני גאה בעצמי על התהליך שעברתי ועל התוצר המרשים שנוצר מעשרות שעות של עבודה קשה.

כתיבת הפרויקט פיתחה אותי כבן אדם. אחרי ביצוע פרויקט בסדר גודל כמו זה, אני מרגיש שעם מספיק מוטיבציה ורצון אין אתגר שאני לא יכול לעמוד בו או פרויקט שאני לא יכול לבצע.

8. הוראות התקנה ותפעול

8.1. תצורה ודרישות קדם

למחשב ה ADMIN:

- נדרש net framework.
- נדרש Windows forms.

למחשב ה Client:

- נדרשים ספריות: scapy, sqlite3,threading,os,sys,time
- פייתון גרסה 3.4 ומעלה.
- מערכת הפעלה Windows 7 ומעלה.
- מומלץ: WireShark

למחשב ה Server:

- נדרשים ספריות threading,sys,time
- פייתון גרסה 3.4 ומעלה.

8.2. התקנה

לאחר התקנות המודלים והתוכנות. תחילה יש להפעיל את השרת, לאחר מכן את ה ADMIN ואז לחבר את כמות הלקוחות הרצויים (לפי כמות הרשתות הפרטיות שרוצים להקים במערכת). מכאן התוכנה תרוץ ותיתן הוראות בעצמה.

9. ביבליוגרפיה

במהלך כתיבת הפרויקט הסתמכתי על מספר מקורות מידע:

- https://www.netacad.com/courses/packet-tracer
- https://www.geeksforgeeks.org/routing-information-protocol-rip/
- https://www.ibm.com/docs/en/i/7.1?topic=routing-information-protocol
- https://www.ibm.com/support/pages/what-icmp-redirect-message
- https://stackoverflow.com/
- https://docs.microsoft.com/en-us/dotnet/api/
- https://www.khanacademy.org/computing/computers-and-internet/xcae6f4a7ff015e7d:the-internet/xcae6f4a7ff015e7d:transporting-packets/a/transmission-control-protocol--tcp

10. נספחים

הצעת פרויקט:

https://docs.google.com/document/d/1j67-YilW1-JAzOQoCz3qvKQgGDPzm0Z4FoXl9aChyVs/edit?usp=sharing