**שרת פרוקסי HTTP/S – סינון אתרים - SafeProxy:**

**פירוט קצר על הפרויקט:**

הפרויקט מיישם שרת פרוקסי עבור פרוטוקולי האינטרנט HTTP, HTTP. באמצעותו יוכלו לוקחות לחסום אתרים מסוימים לבחירתם, להסתיר את הIP שלהם, ולהיות מוגנים מפני אתרים זדוניים.

**הגדרת אופן פעולת התוכנה:**

לקוחות יוכלו להתחבר לשרת דרך GUI – תוכנה על המחשב. התוכנה תציג מסך ראשוני של LOGIN\SIGNUP, שבו יוכלו לקוחות להירשם\להיכנס על מנת לשמור הגדרות מסוימות. יהיה קיים גם USER ADMINISTRATOR – שישלוט על שרת הפרוקסי (מספר פורט, מתי הוא ירוץ).

על השרת קודם כל להאזין לחיבורים, ואז לסווג את בקשותיהם לפי:

* HTTP
* HTTPS – CONNECT (tunnelling)
* HTTPS – CONNECT (TLS-terminated by proxy)

לכן אנחנו צריכים מחלקה שמאזינה לכל הבקשות המגיעות, ממיינות אותן על פי סוגן, ומנתבת למחלקה הנכונה. נקרא לה:

PROXY\_LISTENER

לאחר מכן, לכל אחד מהחיבורים האלה יש דרך פעולה שונה ולכן לכל אחד מהם תהיה מחלקה שונה:

* HTTP – HTTP\_HANDLER
* HTTPS – CONNECT (tunnelling) – HTTPS\_TCP\_TUNNEL\_HANDLER
* HTTPS – CONNECT (TLS-terminated by proxy) – HTTPS\_TLS\_INTERCEPTION\_HANDLER

עבור האופציה האחרונה גם נדרשות 2 מחלקות. האחת - שתתמודד עם TLS. והשנייה - שתייצר תעודות אבטחה, על פי רשות אבטחה שתוקם ותאושר אצל הלקוח. נקרא להן:

TLS\_HANDLER

CERTIFICATE\_HANDLER

כמו כן, על מנת שיהיה ממשק משתמש גרפי תהיה מחלקת:

GUI

חוץ מזה, נדרש מסד נתונים, על מנת לשמור, לערוך ולגשת, לשמות משתמש, סיסמאות, והגדרות שמורות של לקוחות, ושל משתמש העל. זוהי תהיה מחלקת:

DB\_HANDLER

להמשך, אם יתאפשר להוסיף כאשינג, נצטרך מחלקה שתעבוד על פי חוקים מסוימים של שמירה זמנית, תוך שמירה על פרטיות הלקוח – מחלקת:

CACHING

כמו כן, למטרת בדיקות תקינות, ודיבאג, תהיה מחלקת:

UNIT\_TESTS

**מחלקות:**

* ProxyListener
* HttpHandler
* HttpsTcpTunnelHandler
* HttpsTlsInterceptionHandler
* TlsHandler
* CertificateHandler
* GUI
* DBHandler
* Caching

**מחלקות (debugging):**

* UnitTests

**הבדל בין 2 סוגי בקשות הHTTPS:**

Tunnelling (CONNECT)

Client ----HTTP CONNECT----> Proxy

Client <---200 Established---- Proxy

Client ---TLS handshake (encrypted bytes) ---> Proxy (forwards) ---> Server

Client <---Encrypted TLS---> Proxy (forwards) <---Server

Proxy never decrypts.

MITM (TLS-terminating proxy)

Client ---HTTP CONNECT---> Proxy (or transparent interception)

Client <---200 Established-- Proxy

Client ---TLS handshake---> Proxy (Proxy presents a cert for example.com signed by internal CA)

Client <---TLS handshake--- Proxy

Client ---HTTP over TLS (plaintext at proxy) --> Proxy (decrypts)

Proxy ---TLS handshake---> Server (new separate TLS)

Proxy <---TLS handshake--- Server

Proxy ---HTTP over TLS--> Server (re-encrypts)

Proxy sees plaintext and can inspect/modify.