



Bulldog™

What's yours, belongs to you.

תיק פרויקט

פרויקט גמר במסגרת מגמת סייבר

שם התלמיד: עמרי לוי

ת"ז: 318919867

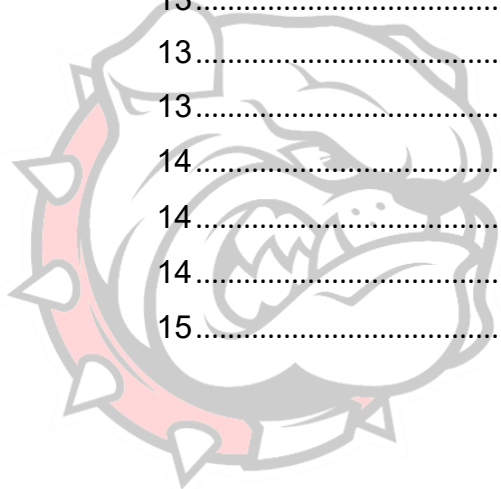
מנחים: מיכאל צ'רנובילסקי וינאי סנד

בית הספר: תיכון הראשונים

מאי 2017

תוכן עניינים

3	תקציר כולל ורציונל הפרויקט
3	רציונל הפרויקט
3	המוטיבציה לפיתוח
4	מבוא ורקע כללי
6	תיאור המוצר המוגמר
7	שפת התכנות וסביבת העבודה
8	ניסוח וניתוח הבעיה האלגוריתמית
8	נעילה מלאה, אמינה ומאובטחת של הקבצים
8	מימוש שרת מרובה לקוחות (multi-client)
8	ממשק משתמש פשוט, מהיר וחלק
9	ניהול של מפתחות ההצפנה
10	תיאור האלגוריתמים הקיימים
10	נעילה ופתיחה של קבצים (הצפנה ופיענוח)
10	מימוש ממשק המשתמש (GUI)
10	מימוש שרת מרובה לקוחות (multi-client)
10	מימוש פרוטוקול העברת המידע
11	מימוש תקשורת מוצפנת
11	שמירת הנתונים עבור הקבצים הנעולים והמשתמשים
12	הפתרון הנבחר
12	נעילה מלאה, עמידה ומאובטחת של הקבצים
12	מימוש שרת מרובה לקוחות
12	ממשק משתמש פשוט, מהיר וחלק
12	ניהול של מפתחות ההצפנה
13	תיאור הרכיבים המרכיבים את המוצר
13	צד הלקוח
13	תהליך ה-GUI
14	מודול הרשת- Networking
14	מודול ההצפנה\ נעילה- Encryption
14	התהליך הראשי- Main Client
15	צד השרת



15.....	מודול התקשורת- Networking
15.....	תהליך מסד הנתונים- Database Operator
15.....	התהליך הראשי- Main Server
16.....	פירוט מבני הנתונים
17.....	פירוט מאגר המידע
17.....	טבלת המשתמשים
17.....	טבלת הקבצים
18.....	השוואת העבודה עם פתרונות ויישומים קיימים
18.....	WinRAR
19.....	Windows
20.....	מדריך ההתקנה למשתמש
21.....	תיאור ממשק המשתמש
21.....	נעילת קובץ- הצפנה
23.....	פתיחת קובץ- פענוח
24.....	מבט אישי על העבודה ועל תהליך הפיתוח
26.....	ביבליוגרפיה
26.....	שימוש בממש הגרפי
26.....	שימוש בהצפנה
27.....	נושאים נוספים



תקציר כולל ורציונל הפרויקט

רציונל הפרויקט

בעולם המודרני של ימינו, קיים שימוש רחב במחשבים הפרטיים, ואנו זקוקים להם יותר ויותר. הם הפכו לשיטה היעילה, החכמה והמקובלת ביותר לאחסון מידע מכל סוג. אולם, מצב זה מעלה בעיה חדשה באורח חיינו- מה אם נרצה מידע סודי או אישי? היום, הגישה למידע הפכה לחופשית וקלה להרבה יותר. אמנם יש בכך הקלה, והמידע החשוב נגיש בזמן אמת, אבל עקב זאת, קשה מאוד למנוע מאנשים את הגישה למידע רגיש, סודי או אישי. כל אחד שרוצה את המידע יכול לגשת למחשב ולקחת אותו בעזרת אחסון נייד. גם אם מחשב זה נעול בעזרת סיסמא, קיימות עוד מגוון דרכים נוספות לגשת למידע ולקחת קבצים מהמחשב האישי.

Bulldog™ פותח על מנת להגן על המידע והקבצים באופן מלא, גם אם הם דלפול נגנבו, בזמן שלמורשים, ניתן לגשת למידע בכל זמן ומכל מקום.

המוטיבציה לפיתוח

המוטיבציה לפיתוח נבעה מהצורך הברור להגן על המידע האישי והפרטי של בעלי המחשבים האישיים. הסיכון של גניבת מידע סודי או חשיפת מידע אישי הוא אמיתי, וכל אחד מאתנו נתון תחת סיכון זה כל הזמן. לכן, היה לי חשוב ליצור כלי שיפתור את הבעיה הזאת. בנוסף, לדעתי, לא קיים כלי שמאפשר לאבטח את המידע ולנעול אותו בצורה קלה ונגישה כמו Bulldog™. הנגישות והנוחות הן המפתח, הן מאפשרות לכל אדם לשמור על המידע שלו סודי ונעול, בכל זמן, ומבלי להסתבך.



מבוא ורקע כללי

המטרה המרכזית של המוצר היא למנוע מחשיפה של מידע אישי, סודי או רגיש מתוך המחשבים האישיים. זאת על ידי הצפנה של הקבצים, ושמירה של מפתחות ההצפנה על שרת מרוחק, כך שניתן לגשת אליהם מכל מחשב, אך לא ניתן לגנוב אותם.

המערכת מצפינה את תוכן הקבצים, בהתאם לשיטת ההצפנה שבחר המשתמש (כאשר המערכת מסבירה לו כיצד לבחור בצורה הנכונה). לאחר שהמשתמש התחבר למערכת בעזרת שם משתמש וסיסמא, היא מצפינה את הקובץ התיקיה, (כאשר היא מצפינה תיקיה, כל קובץ בתיקיה מקבל מפתח הצפנה שונה) ושומרת את מפתחות ההצפנה על השרת המרוחק, כדי למקסם את האבטחה. בנוסף, התקשורת בין השרת ללקוח מוצפנת, ובכך המערכת לא מאפשרת גניבה של סיסמאות או מפתחות הצפנה.

התרחישים שהמערכת צריכה להתמודד איתם:

- רישום והתחברות של משתמש (ניהול מאגר מידע שכולל את המשתמשים ואת מפתחות ההצפנה שלהם)
- נעילה של קובץ (הצפנה של קובץ)
- נעילה של תיקיה (נעילה של כל הקבצים בתיקיה וכל הקבצים בתת-התיקיות שבה)
- פתיחה של קובץ (פענוח של קובץ)
- נעילה של תיקיה (נעילה של כל הקבצים בתיקיה וכל הקבצים בתת-התיקיות שבה)

הבעיות שהמערכת צריכה לדעת להתמודד איתן:

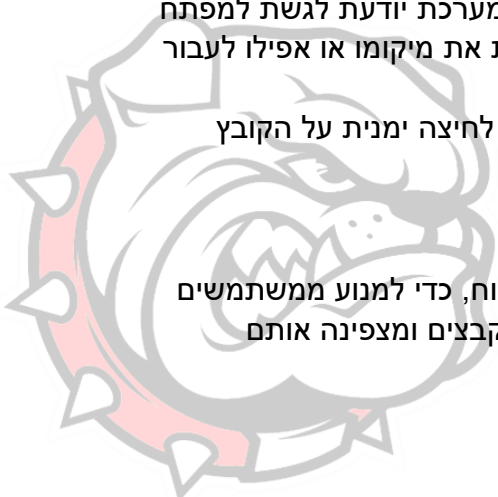
- 1) מפתחות ההצפנה צריכים להיות שמורים במקום שאינו נגיש לאף משתמש שאינו בעל ההרשאה המתאימה, בעוד שמשתמשים בעלי ההרשאה המתאימה יוכלו לגשת לפרויקט מכל מקום.
- 2) הקובץ המוצפן עלול להיגנב\ לעבור למיקום שונה במחשב
- 3) המערכת צריכה להיות נוחה ופשוטה ככל האפשר, כדי לאפשר לכל אדם לנעול (להצפין) את הקבצים שלו בקלות.

המערכת מתמודדת עם בעיות אילו באופן הבא:

- 1) המערכת שומרת את מפתחות ההצפנה בשרת מרוחק, ולא קיים סיכון שתוקף ימצא את מפתח ההצפנה של הקובץ במחשב או בקובץ עצמו.
- 2) המערכת יודעת לפענח או להצפין כל קובץ, גם אם הוא עבר מקום או אפילו מחשב. המערכת שומרת מידע סודי בקובץ, שמתאר את מיקום המפתח שלו במאגר הנתונים, במקום לשמור במאגר הנתונים את מיקום הקובץ במחשב או את השם שלו. כך, המערכת יודעת לגשת למפתח מבלי לשמור אותו במקום הנגיש לתוקפים בעוד שהקובץ יכול לשנות את מיקומו או אפילו לעבור בין מחשבים.
- 3) ממשק המשתמש הוא נוח ופשוט, וניתן להשתמש במערכת בעזרת לחיצה ימנית על הקובץ שנבחר.

הפרויקט משלב מספר תחומים:

- תקשורת ברשת- המערכת צריכה את מפתחות ההצפנה במקום בטוח, כדי למנוע ממשתמשים ללא הרשאות לגשת למפתחות. לצורך כך, המערכת שנועלת את הקבצים ומצפינה אותם



מתקשרת עם שרת מרוחק שמאחסן את מפתחות ההצפנה. גם התקשורת מוצפנת, והיא נעשית באמצעות שימוש בפרוטוקול BDTP (Bulldog™ Data Transfer Protocol) שבניתי במהלך העבודה על הפרויקט. פרוטוקול זה בנוי על הפרוטוקולים TCP/IP, אשר מאפשר אימות של המידע אשר נשלח. הוא מוודא שכל המידע שנשלח מתקבל בסדר הנכון ודואג לשלוח אותם שוב אם לא.

- תקשורת בין תהליכים (IPC- Inter-Process Communication) מתרחשת תקשורת בין תהליך המנהל את ממשק המשתמש לבין תהליך ראשי המנהל את התקשורת עם השרת המרוחק ואת הנעילה של הקבצים.
- עבודה עם מערכת ההפעלה- כדי להבטיח את חווית המשתמש הטובה והנוחה ולהביא לנעילה מלאה של הקבצים, המערכת ניגשת להגדרות מערכת ההפעלה השמורות ב-registry.
- הצפנה ואבטחת מידע- תחום זה משולב בכמה דרכים בפרויקט. ראשית, הנעילה של הקבצים עליהם המשתמש מעוניין להגן מתבצעת באמצעות הצפנה של תוכן הקובץ. המערכת משתמשת בכמה סוגים של הצפנה, כאשר המשתמש יבחר בשיטה האופטימלית עבורו. בנוסף, התקשורת מתרחשת על ערוץ מוצפן והחלפת הסיסמאות נעשית באופן מוצפן.
- ניתוח וכתובה של קבצים- המערכת מצפינה את הקבצים, ומוסיפה בראשם מספר header שמספקים מידע על הקובץ ועל אחסון המפתחות בשרת. המערכת כותבת קבצים כאשר היא מצפינה, והיא מנתחת את header השונים בקובץ כאשר היא מפענחת.
- ממשק משתמש גרפי- המערכת משלבת ממשק גרפי למשתמש, נוח לשימוש, פשוט ומגייב.



תיאור המוצר המוגמר

שם המוצר: Bulldog™

תיאור המוצר: המוצר מאפשר למשתמשים לנעול את הקבצים האישיים החשובים להם, המכילים מידע סודי או אישי, על ידי הצפנה. כמו כן, המוצר מאפשר ריבוי של משתמשים על אותו מחשב, ומאפשר תמיכה בקבצים שמיקומם בזיכרון שונה או אפילו עברו למחשב אחר.

מטרת המוצר: המטרה המרכזית של המוצר היא לאפשר למשתמשים לשמור את המידע הסודי, הפרטי או האישי שלהם על המחשב האישי שלהם, מבלי לחשוש שמשתמש אחר ינסה לגשת למידע שלהם. בנוסף, המטרה היא לאפשר נעילה זו באופן נוח ופשוט, שיאפשר לכל משתמש לשמור על המידע החשוב לו, ללא קשר לידע הקודם שלו בשימוש במחשב.

השאלות שהמוצר עונה עליהן

- 1) כיצד משתמש יכול לחסום קבצים?** ניתן לחסום\ לנעול קבצים על ידי הצפנה- המערכת מצפינה את תוכן הקובץ, ובכך הופכת את הקובץ לחסר כל ערך ולא קריא עבור כל תכנית או משתמש שאינו בעל מפתח ההצפנה.
- 2) כיצד לאפשר למשתמש לגשת לקבצים שלו מבלי שיצטרך לזכור את מפתחות ההצפנה שלו?** את מפתחות ההצפנה ניתן לשמור ולהקביל אותן למשתמש וסיסמא- כך, כל משתמש זוכר את שם המשתמש שלו ואת הסיסמא שלו בלבד, ומיד מקבל גישה לכל מפתחות ההצפנה שלו, בלי שהוא צריך לזכור אותם.
- 3) כיצד להגן על מפתחות ההצפנה של הקבצים?** שכן משתמשים שונים תמיד ינסו לגנוב מידע ואת מפתחות ההצפנה. לכן, צריך למצוא מקום שאינו נגיש כדי לשמור בו את המפתחות. המערכת עונה על שאלה זו בכך שהיא שומרת את מפתחות ההצפנה בשרת מרוחק, תוך שימוש בתקשורת מוצפנת, ובכך מונעת ממשתמשים לא מורשים את הגישה למפתחות, לעומת מערכות שמנסות לשמור את מפתחות ההצפנה במקום פגיע בתוך הקובץ או במחשב.
- 4) כיצד להתמודד עם העברה של קבצים חסומים?** המערכת שומרת מידע על איפה נשמר מפתח ההצפנה בשרת בקובץ, אך לא על מפתח ההצפנה עצמו. כך, גם אם משתמש לא מורשה הצליח למצוא את המידע הזה, הוא חסר ערך עבורו, שכן אין לו גישה למפתחות. אולם, המידע תמיד נשאר עם הקובץ, וכך גם אם הוא מועבר, מיקומו של מפתח ההצפנה אינו משתנה והגישה אליו אינה משתנה.
- 5) כיצד להגן על התעבורה בין השרת ללקוח ולמנוע גניבה של מפתחות או סיסמאות?** השרת והלקוח מצפינים את התקשורת ביניהם, ובכך מונעים מכל ישות המאזינה לרשת לגנוב את המידע שעובר ביניהם- את הסיסמאות או המפתחות. בנוסף, הסיסמאות מועברות לאחר שעברו בפונקציה חד- כיוונית (hash). המשמעות היא שהן הופכות לstring אחר, שאותו לא ניתן לתרגם חזרה לסיסמא האמיתית. לעומת זאת, מחרוזת זו שמורה בשרת, והוא יכול לבדוק אם הסיסמא נכונה או לא. כך, גם אם הסיסמאות נגנבו, לא יהיה ניתן למצוא את הסיסמאות האמיתיות.



שפת התכנות וסביבת העבודה

הפרויקט כתוב בשפת Python, עם מעט חלקים של SQL לניהול מאגר המידע. הסיבה לכך היא ש-Python היא שפה נוחה לשימוש ועונה על כל צרכי המערכת, גם בתור שרת וגם בתור לקוח. לצורך מימוש הממשק הגרפי, השתמשתי בחבילה PyQt4, שמבוססת על חבילת הגרפיקה בשפת C++, החבילה Qt.

הכתיבה של הקוד של Python הייתה ב-PyCharm. זהו IDE אשר מאפשר עבודה מאוד נוחה עם קוד Python. היא מאפשרת עבודה על פרויקטים עם מודולים וקבצים רבים בצורה פשוטה ונוחה לתפעול, ולכן השתמשתי בה לכתיבת המודולים השונים ואף לכתיבת הקוד הסופי. בנוסף, היא תומכת בשימוש בתכניות למעקב גרסאות (git, mercurial וכו'...). יתר על כן השתמשתי בה על מנת להריץ ולדבג את הקוד.

עקבתי אחרי גרסאות הפרויקט בעזרת git והשתמשתי בענן של github כדי לגבות תמיד את הפרויקט שלי. השתמשתי בשיטה זו כי היא נוחה לשימוש ולהבנה, מאפשרת המון צרות של עבודה, נפוצה ונתמכת על ידי PyCharm.

את ממשק המשתמש עיצבתי באמצעות סביבת עבודה המעצבת את הממשק בעורך ויזואלי (PyQtDesigner) ומתרגמת אותו לקוד ב-Python. המעצב הגרפי הוא נוח לשימוש ומתרגם את העיצוב ישירות לקוד.



Code less.
Create more.
Deploy everywhere.



ניסוח וניתוח הבעיה האלגוריתמית

במהלך העבודה על הפרויקט, נתקלתי בכמה קשיים ובעיות אלגוריתמיות אותן הייתי צריך לפתור:

נעילה מלאה, אמינה ומאובטחת של הקבצים

בסופו של דבר, מטרת המוצר היא לנעול את הקבצים ולהבטיח שרק המשתמש שנעל אותם יכול לגשת אליהם ולמידע שבהם בלבד. לכן, קיים הצורך להשתמש באבטחה הטובה והאמינה ביותר שניתן, תוך שילוב של מספר שיטות אבטחה, ולדאוג שבאמת אי אפשר לפענח את הקבצים או לשחזר אותם ללא ההרשאות. כדי לעשות זאת, יש להתחשב בשיקולים הבאים:

- **שמירה של מפתחות ההצפנה** - יש לשמור את מפתחות ההצפנה של הקבצים במקום שאינו נגיש למשתמשים בעלי כוונות זדוניות, ועם זאת נגיש מכל מקום בו נמצא הקובץ.
- **העברה של קובץ** - המערכת צריכה להתמודד עם מקרה בו קובץ שינה את מיקומו, או אפילו נגנב מהמחשב והועבר למחשב אחר. יש לדאוג שהמידע אינו נאבד ועדיין ניתן לפיענחו, ובנוסף שגם מחוץ למחשב בו הוא ננעל, הוא נשאר נעול והגישה אליו עדיין חסומה למשתמשים שאינם בעלי הסיסמא.
- **נעילה חכמה של קובץ** - יש להשתמש בשיטת ההצפנה המתאימה לצורך ההצפנה, בהתאם לסוג ולגודל הקבצים שננעלים.

מימוש שרת מרובה לקוחות (multi-client)

אחת מן הבעיות הייתה יצירת שרת אשר מסוגל לספק שירות למספר משתמשים במקביל, מבלי שתורגש הפרעה אצל אף אחד מן המשתמשים הללו. זאת אומרת, השרת צריך במקביל גם להאזין לחיבורים נכנסים וגם לספק מענה לבקשות המשתמשים המחוברים. כדי לפתור בעיה זו, יש לפתור עוד כמה סוגיות נוספות:

- **אופן הטיפול בלקוחות** - כנאמר לעיל, השרת צריך להיות מסוגל להתמודד עם מספר פניות במקביל ואף לטפל בכל הלקוחות אשר כבר מחוברים אליו. לכן צריך למצוא פתרון אשר יאפשר מימוש של שרת כזה, שכן שרת בסיסי לא מסוגל לבצע פעולות שכאלו.
- **בעיית אבטחה** - השרת יכול לשלוח מידע רגיש ללקוחות ולכן חשוב שהמידע שנשלח יהיה מוצפן. זאת כדי שאף אדם לא יוכל לגלות מה היה המידע שנשלח אל הלקוח מלבד הלקוח עצמו. הבעיה האלגוריתמית היא הצפנת המידע בצד השולח ופיענוחו בצד המקבל.
- **פרוטוקול בין השרת ללקוחות** - השרת צריך לדעת איך לתקשר עם הלקוחות, זאת אומרת, בהינתן בקשה מסוימת הוא צריך לדעת כיצד לפעול ואיזה מענה להחזיר. על מנת לפתור בעיה זו פיתחתי פרוטוקול תקשורת אשר מגדיר את צורת ההידברות של הלקוח עם השרת ולהפך.

ממשק משתמש פשוט, מהיר וחלק

מטרת המערכת היא לספק מענה לשמירת המידע לכל משתמש. על כן, קיים צורך בממשק משתמש נוח לשימוש, מגיב ופשוט להבנה, שעובד באופן חלק וללא בעיות.



ניהול של מפתחות ההצפנה

המערכת צריכה לנהל את כל מפתחות ההצפנה, של כל הקבצים וכל המשתמשים, לדעת להפריד ביניהם ולשלוח אותם ללקוחות באופן חכם ויעיל בכל פעם שלקוח מורשה מבקש מפתח.



תיאור האלגוריתמים הקיימים

נעילה ופתיחה של קבצים (הצפנה ופיענוח)

כדי לאפשר את השימוש הנוח ביותר במודול ההצפנה, מודול זה מספק שתי פעולות מרכזיות, כאשר אחת נועלת קובץ\ תיקייה והשנייה פותחת אותם (לאחר שמקבלת את מפתח ההצפנה). פעולות אלה חוסכות למתכנת המשתמש בהן התעסקות רבה.

פעולת ההצפנה תחילה בונה את האובייקט המצפין טקסט (מתוך החבילה PyCrypto). היא פותחת את הקובץ אותו היא רוצה להצפין כקובץ המקור, ופותחת גם קובץ חדש כקובץ היעד, עם סיומת הקובץ הנעול של Bulldog™. לאחר מכן, היא קוראת מספר בלוקים מקובץ המקור, מצפינה אותם בעזרת האובייקט שיצרה ושומרת אותם בקובץ היעד. לבסוף, היא תסגור את קובץ היעד ותמחק את קובץ המקור.

פעולות הפיענוח עובדת באופן זהה.

מימוש ממשק המשתמש (GUI)

כל חלון של ממשק המשתמש רץ בתהליך נפרד. כך, הוא אף פעם לא עובר למצב מגיב. התהליך הראשי של הלקוח פותח את התהליכים השונים עבור החלונות, ומתקשר איתם בעזרת pipe.

בנוסף, החלונות השונים מתקשרים אם התהליך הראשי רק כאשר יש להם מידע להעביר אליו. הם עושים זאת באמצעות שימוש בsignals ו-slots- הם מאותתים לפעולה לפעול רק כאשר יש תהליך שצריך להתבצע. גם כאשר התהליך מתבצע, הוא מנותק מהגרפיקה ומתבצע בthread שונה, ובכך מאפשר לחלונות להמשיך לעבוד באופן רציף.

מימוש שרת מרובה לקוחות (multi-client)

כדי לממש את השרת מרובה הלקוחות, השתמשתי במודול threading ו-select בפייתון. השרת רץ בלולאה תמידית אשר מקבלת חיבורים מלקוחות שונים. אולם, לפני שהשרת מקבל חיבור, הוא בודק האם ניתן לקבל חיבור חדש מהsocket בו הוא משתמש בעזרת המודול select. כאשר מתקבל חיבור חדש, השרת פותח thread חדש שמקבל את החיבור, מנהל איתו session ומטפל בכל הבקשות שלו.

מימוש פרוטוקול העברת המידע

כדי שהמערכת תעבוד בצורה הטובה ביותר, בחרתי לבנות פרוטוקול משלי להעברת המידע בין השרת ללקוחות. BDTP- Bulldog™ Data Transfer Protocol. הפרוטוקול בנוי בצורה הבאה:

	0	1	2	3	4	5	6	7	8	9	0A	0B	0C	0D	0E	0F
0	operation	operation	operation	operation	size	size	status	status	data	d	d	d	d	d	d	d

כאשר כל בייט אומר:

- 1) Operation- The type of request the client attempts to perform or the server answers.
The options are: LIN (login), LOU (logout), lock file(ADD), decrypt file (DEC), create

encrypted connection (CON), kickout from sever (KICK) or SUP (signup). Note: if the operation field's value is smaller than 4 bytes, padding will be added.

- 2) Size- The size of the received data (message content).
- 3) Status- The status code of the operation- like HTTP. This field is only used by the server, as the clients always puts 0 under this field (the code which stands for request in the protocol).
- 4) Data- The actual data of the request. For example- the key and iv used to encrypt a certain file.

הפרוטוקול קל לשימוש, אינו מכיל שדות מיותרים ועדיין מכיל את כל השדות ההכרחיים, כך שהתקשורת מתבצעת באופן המהיר והיעיל ביותר.

מימוש תקשורת מוצפנת

כדי לממש תקשורת מוצפנת ומאובטחת, הלקוח מתחיל בכך שהוא שולח מפתח RSA פומבי לשרת בעזרת פרוטוקול BDTTP. השרת יבחר מפתח AES להצפנה סימטרית של התקשורת, וישלח אותו בחזרה ללקוח. הלקוח מפענח את ההודעה בעזרת מפתח ה-RSA הפרטי שלו, וכעת מתחיל להצפין ולפענח את כל התעבורה בעזרת מפתח ה-AES המשותף.

השרת בוחר מפתח AES שונה לכל session שנוצר.

שמירת הנתונים עבור הקבצים הנתונים והמשתמשים

שמירת הנתונים מתבצעת בבסיס נתונים מסוג SQL. אולם, הלקוח לא צריך לדעת להשתמש ב-SQL, ונוסף על כך, אם ללקוח תהיה גישה ישירה לבסיס הנתונים זו תהווה סיכון אבטחה גבוהה. לכן, אחד התפקידים המרכזיים בשרת הוא לתרגם את בקשותיהם של הלקוחות, מהשימוש בפרוטוקול לעדכון בסיס הנתונים בשפת SQL.

בכל session שהשרת מקיים, הוא מקבל את פניית הלקוח ובודק אם אכן יש ללקוח את ההרשאות המתאימות לביצוע הפעולה שהתבקשה. אם כן, השרת יבצע את הפעולה ויעדכן את בסיס הנתונים, בזמן שהוא הופך את בקשת הלקוח ל-SQL Query. השרת לאחר מכן יחזיר ללקוח הודעה המציינת אם הפעולה התבצעה בהצלחה, ואת תשובת הבקשה (אם יש כזאת).

בזמן שהשרת עובד, רץ תהליך נוסף המפעיל שרת SQL על בסיס הנתונים של המערכת. תהליך זה מקיים תקשורת עם המערכת של השרת על loopback, מאזין לכל שאילתות ה-SQL שלו ומבצע אותן.



הפתרון הנבחר

נעילה מלאה, עמידה ומאובטחת של הקבצים

- **אחסון מפתחות ההצפנה על שרת מרוחק**- מפתחות ההצפנה של הקבצים, בהם המערכת משתמשת כדי לנעול את הקבצים, לא מאוחסנים על המחשב בו הקובץ מוצפן, אלא על שרת מרוחק. כך, משתמשים בעלי כוונות זדוניות לא יוכלו לגשת למפתחות, משום שהם כלל לא נמצאים על המחשב.
- **מציאת מפתח שונה עבור כל קובץ**- כל קובץ שמוצפן מקבל מפתח הצפנה אחר. כך, גם אם הייתה פרצת אבטחה כלשהי, ואחד המפתחות נמצא, הוא לא חושף את שאר הקבצים. בנוסף, למשתמשים שמנסים לגנוב מפתחות קשה יותר למצוא את המפתחות הנכונים.
- **שימוש במספר שיטות הצפנה**- המערכת יודעת לנעול את הקבצים תוך שימוש במספר שיטות הצפנה, כאשר כל שיטה תואמת את הקבצים שנבחרו (שיטה אחת טובה יותר לקבצים גדולים, אחת להרבה קבצים וכו'...)

מימוש שרת מרובה לקוחות

השרת שבניתי מצליח להתמודד עם כל אחת מהבעיות שצוינו בצורות הבאות:

- **אופן הטיפול בלקוחות**- השרת פותח יחידת ריצה (thread), עבור כל session של חיבור עם לקוח. כל לקוח מקבל יחידת ריצה, המשאבים של השרת מחולקים באופן שווה בין כל הלקוחות והוא מסוגל לתת מענה למספר לקוחות במקביל.
- **בעיית האבטחה**- קיים הצורך להצפין את התקשורת, כדי למנוע גניבה של סיסמאות או מפתחות הצפנה. לצורך כך, בניתי פרוטוקול שתומך בהצפנה, ואובייקט העוטף את מחלקת socket ומטפל באופן מלא בהצפנה.
- **פרוטוקול בין השרת ללקוח**- השרת צריך לדעת איך לתקשר עם הלקוחות, זאת אומרת, בהינתן בקשה מסוימת הוא צריך לדעת כיצד לפעול ואיזה מענה להחזיר. על מנת לפתור בעיה זו פיתחתי פרוטוקול תקשורת אשר מגדיר את צורת ההידברות של הלקוח עם השרת ולהפך.

ממשק משתמש פשוט, מהיר וחלק

כדי ליצור ממשק משתמש מגיב, מהיר ופשוט לתפעול, יצרתי תת-תהליך של GUI שמטרתו להתנהל מול המשתמש בלבד, לענות לבקשותיו ולשלוח אותן לתהליך הראשי, המטפל בנעילת הקבצים ובתקשורת מול השרת. תהליך ה-GUI שולח את בקשות המשתמש בעזרת Pipe.

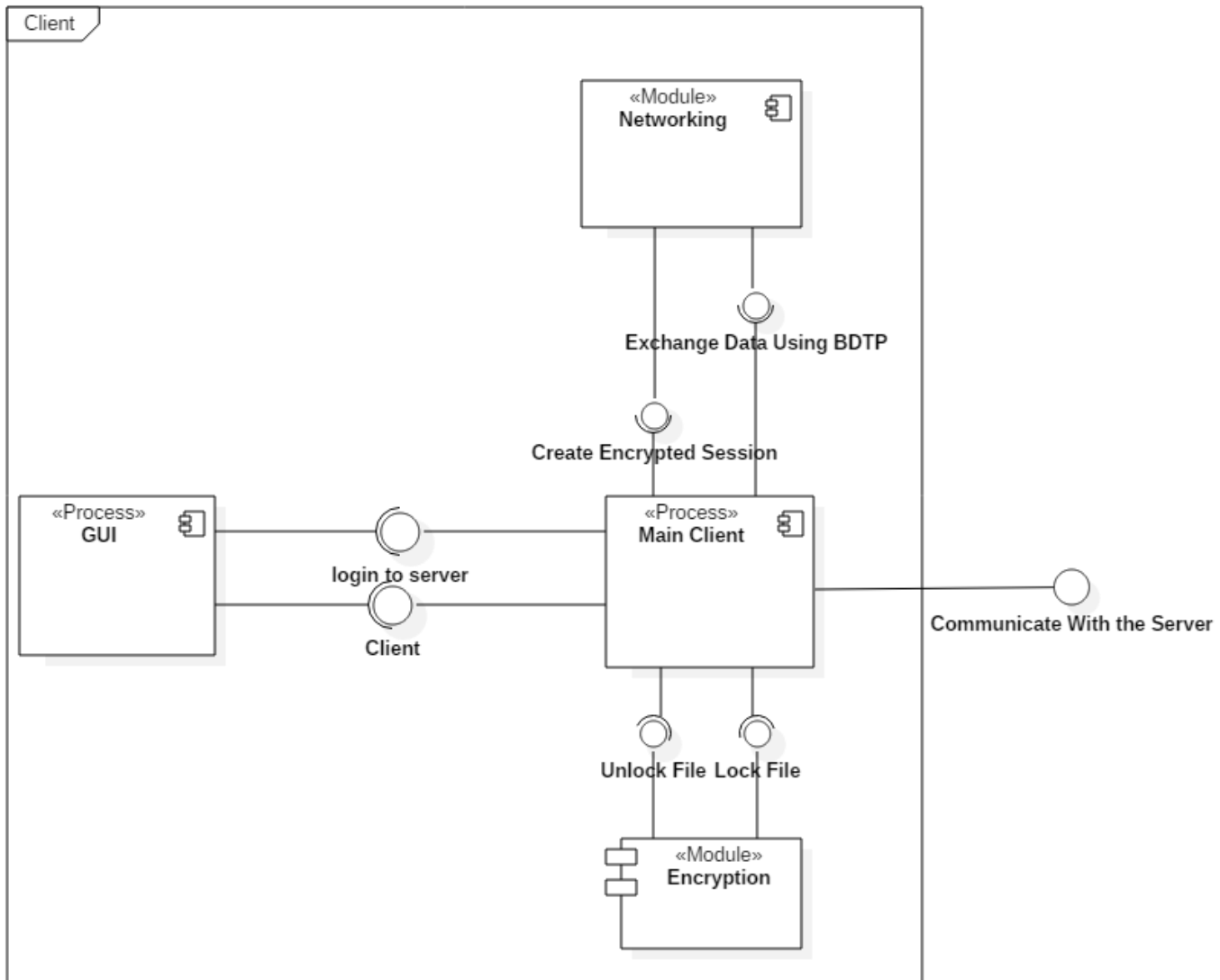
ניהול של מפתחות ההצפנה

השרת שומר את מפתחות ההצפנה באמצעות מאגר נתונים של SQL, כאשר טבלה אחת שומרת את המשתמשים ואת המידע עליהם, בעוד שטבלה נוספת שומרת את הקבצים ומפתחות ההצפנה. השרת קושר בין הטבלאות, ובכך יודע לנהל בצורה חכמה ויעילה את מפתחות ההצפנה ולספק שירות יעיל ומהיר ללקוחות.



תיאור הרכיבים המרכיבים את המוצר

צד הלקוח



צד הלקוח מורכב מהרכיבים הבאים:

תהליך ה-GUI

תהליך ממשק המשתמש הוא תהליך שרץ בנפרד מהתהליך הראשי, הוא ילד של התהליך הראשי. תפקידו הוא לטפל בבקשות של המשתמש- להתחבר עם הסיסמא שלו, לנעול או לפתוח קבצים ועוד...

הסיבה לכך שזהו תהליך נפרד היא, שכאשר ה-GUI נמצא באותו תהליך עם המנוע של התכנית, עולות בעיות רבות- בכל ביצוע של פעולה חיושית מסובכת (כמו הצפנה של קובץ, למשל), פנייה לאינטרנט או כל תהליך blocking אחר, תהליך ה-GUI יעבור למצב "לא מגיב". בשלב זה, המשתמש הנורמלי מנסה

לסגור את התהליך בכל מחיר ולצאת מהתכנית שקפאה. לעומת זאת, כאשר GUI מנוהל בתהליך נפרד, בעיה זו נפתרת, שכן התהליך כולו עוסק רק בGUI ובתקשורת עם התהליך הראשי, כך שאינו קופא ומגיב כל הזמן למשתמש.

בנוסף, ברוב חבילות GUI השונות, קיימת בעיה לשנות אלמנטים גרפיים מחוץ לthread הראשי. המשמעות היא שלא ניתן להשתמש בthread שונים, וקשה לעבוד באופן יעיל ורציף. לעומת זאת, כאשר הפעולות המסובכות והגישות לשרת מתנהלות מתהליך אחר, בעיה זו נפתרת.

תהליך זה תלוי בתהליך הראשי כדי להתחבר בתור משתמש לשרת או כדי להצפין\לפענח קבצים, ואילו הן פעולותיו העיקריות.

מודול הרשת- Networking

מודול זה מספק לתהליך הראשי את ה-interface הדרוש לקיום תקשורת מוצפנת התואמת את הפרוטוקול של Bulldog™. הוא עוסק את מחלקת socket בפייתון, ובכך מאפשר שימוש, נוח טבעי ונכון לתקשורת.

בנוסף, מודול זה מספק interface נוסף שמאפשר לנתח את השדות השונים בפרוטוקול בקלות ולגשת אליהם בנפרד. בכך, הוא הופך את השימוש בפרוטוקול לטבעי ומוודא ששני הצדדים בתקשורת משתמשים בפרוטוקול בצורה הנכונה.

מודול ההצפנה נעילה- Encryption

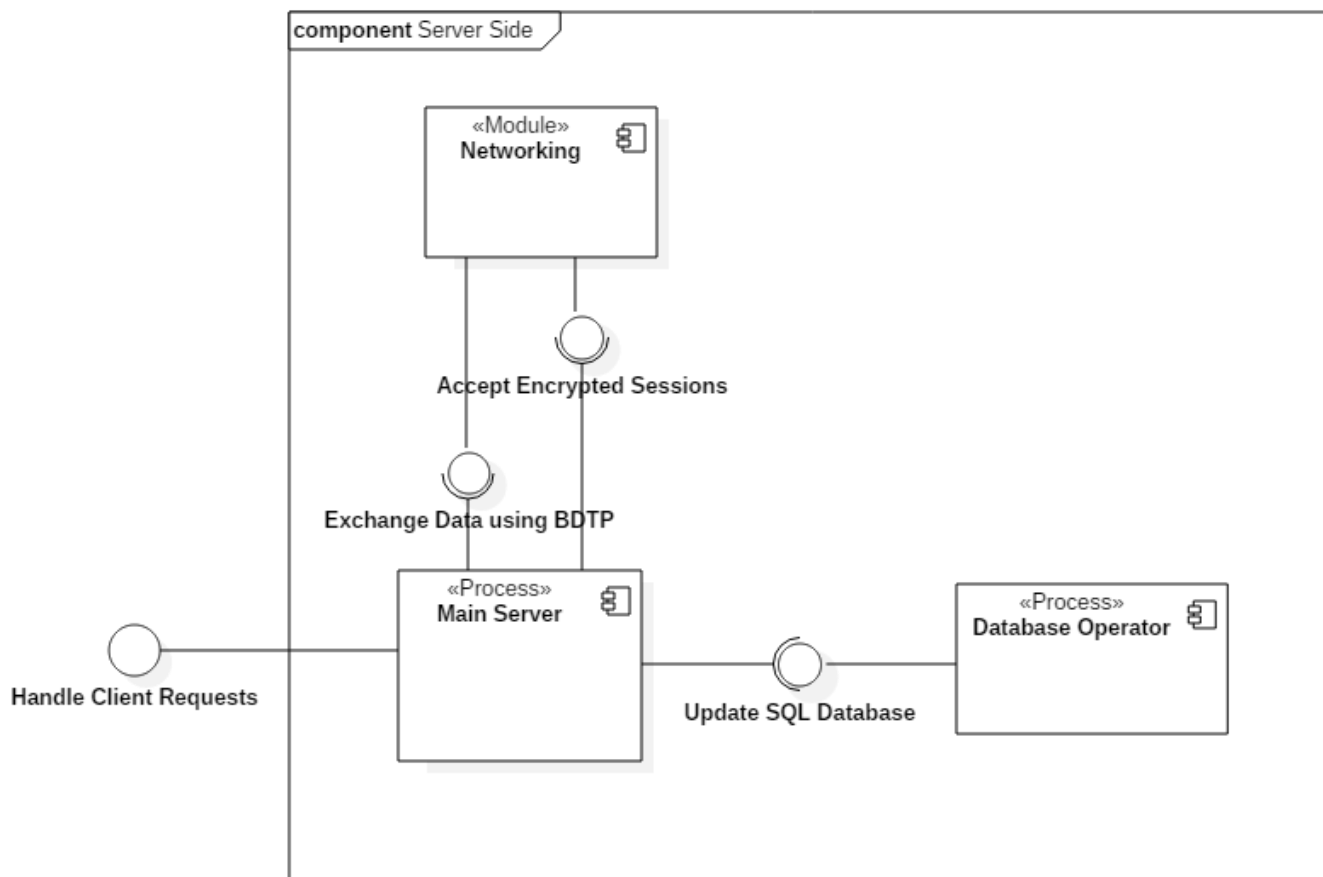
מודול זה מכיל את הפעולות הנחוצות להצפנה\נעילה של קבצים ותיקיות. הוא מצמצם את כל האובייקטים והפעולות שבתוך המודול לשתי פעולות מרכזיות- encrypt_path ו-decrypt_path- כלומר, כל מה שנדרש הוא לספק לפעולות אילו את הקבצים במחשב שברצוננו לנעול, והמודול ידאג לנעול אותן בעזרת שיטות ההצפנה השונות תוך שימוש במפתחות שונים.

תהליך הראשי- Main Client

זהו התהליך שמחבר בין כל הרכיבים האחרים. התפקיד שלו הוא לנהל את הבקשות המתקבלות מתהליך הGUI, לתקשר עם השרת בעזרת הפרוטוקול כדי לענות על הבקשות, ולאחר מכן להציג אותן חזרה בתהליך הGUI.

התהליך הראשי עושה זאת תוך שימוש במודול ההצפנה כדי לבצע את נעילת הקבצים, ותוך שימוש במודול התקשורת כדי לקיים תקשורת מאובטחת ונכונה (העומדת בפרוטוקול) עם השרת.





מודול התקשורת- Networking

מודול זה זהה למודול התקשורת בצד הלקוח. כפי שצוין, המודול מספק אובייקטים וממשקים המאפשרים לקיים תקשורת מוצפנת לפי חוקי הפרוטוקול BDTP, תוך שימוש בממשק טבעי ונוח.

תהליך מסד הנתונים- Database Operator

תהליך שרץ בנפרד מהתהליך הראשי של השרת. תפקידו הוא לתקשר עם התהליך הראשי בשרת, לקבל את הבקשות לעדכן את מסד הנתונים של השרת ואת הבקשות שלו להוציא ממנו מידע. הוא מקבל, דרך ערוץ תקשורת עם השרת, פקודות SQL שעליו לבצע.

תהליך הראשי- Main Server

זהו התהליך הראשי של השרת. הוא מטפל בפניות של כל הלקוחות, בכך שהוא מקיים מספר חיבורים מוצפנים במקביל ומספק שירות לכל המשתמשים המחוברים. הוא מעדכן את מסד הנתונים בהתאם לבקשות של המשתמשים, ודואג שאף משתמש לא יוכל לגשת לקבצים שלא שייכים לו.

פירוט מבני הנתונים

מודול	שם המבנה	סוג המבנה	תפקיד
ServerMain.py	ActiveClient	מחלקה	עבור כל משתמש מחובר קיים אובייקט כזה. מטרתו היא לשמור את כל הפרטים הדרושים עבור כל משתמש מחובר - שם המשתמש, מספר ה-ID שלו, הsocket שבו הוא משתמש והפעם האחרונה שהוא ביצע פעולה (משתמשים לא פעילים מנותקים באופן אוטומטי).
	Logged_in_users	רשימה של ActiveClient	רשימת כל המשתמשים המחוברים לשרת, כאשר לכל אחד מהם נשמר מידע באובייקט ActiveClient
networking.py	EncryptedFile	מחלקה	מחלקה זו מכילה את כל הפרטים על הקובץ הנעול מלבד המידע עצמו (data) שלא צריך לעבור לשרת. כך, השרת והלקוח יכולים להעביר מידע על קבצים נעולים בקלות
	BDTPMessage	מחלקה	מחלקה זו מכילה את כל שדות הפרוטוקול ויכולה להישלח דרך socket בקלות. כך, קל מאוד להשתמש בפרוטוקול BDTP.
	BDTPSocket	מחלקה	זוהי מחלקה שמתלבשת על אובייקט הsocket בפיתוח. הBDTPSocket יוצר חיבור מוצפן באופן אוטומטי, כאשר ממשק השימוש בו זהה לממשק של האובייקט הבנוי בפיתוח. הוא מאפשר שימוש קל וטבעי בפרוטוקול ובחיבור מוצפן.
GUI.py	*Window	מחלקה	המחלקות במודול הממשק הגרפי שנגמרות בWindows מייצגות חלון כלשהו בממשק המשתמש. לכל חלון יש מחלקה משלו, שנוצרה על ידי המעצב של PyQt.

פירוט מאגר המידע

מאגר המידע של המערכת נמצא בשרת, והוא שמור בבסיס נתונים SQL. בבסיס הנתונים קיימות שתי טבלאות:

טבלת המשתמשים

שומרת את כל המידע אודות המשתמשים הרשומים במערכת:

user_id	username	password	full_name	email
INTEGER	STRING	STRING	STRING	STRING

זאת כאשר המספר user_id הוא המזהה החד ערכי של השורות בטבלה, הוא ייחודי לכל משתמש.

טבלת הקבצים

שומרת את כל המידע אודות הקבצים שנעלו על ידי המערכת:

file_id	user_id	method	iv	key	date_encrypted
INTEGER	INTEGER	INTEGER	STRING	STRING	STRING

-user_id המזהה החד ערכי של המשתמש שהצפין את הקובץ.

-file_id מזהה הקובץ. ערך זה אינו חד ערכי, שכן ייתכן שלשני קבצים יהיה אותו מספר id אם הם נעלו על ידי משתמשים שונים.

-method שיטת ההצפנה בה הקובץ ננעל (המערכת מתאימה לכל ערך מספרי את שיטת ההצפנה המתאימה).

-iv ה Initialization Vector של הקובץ המוצפן.

-Key מפתח ההצפנה של הקובץ.

-date_encrypted התאריך והשעה בו הקובץ ננעל. נשמר כ-String.

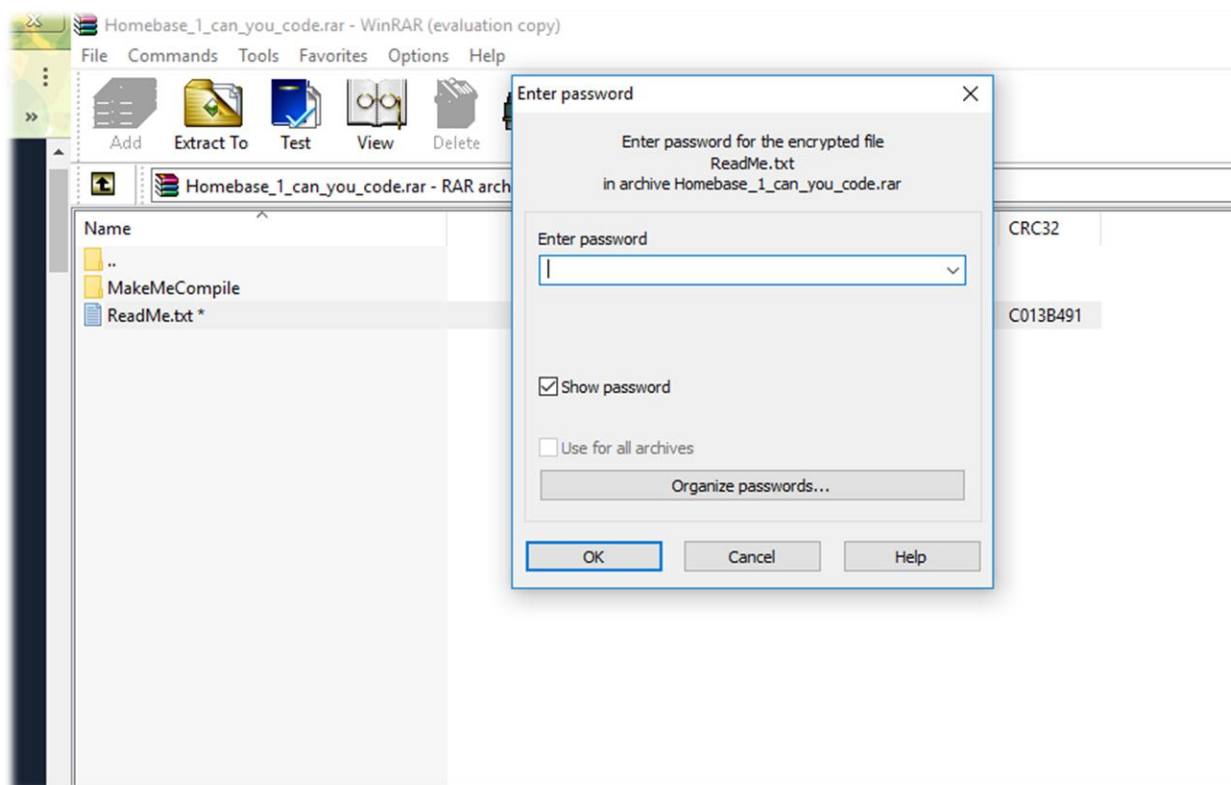


השוואת העבודה עם פתרונות ויישומים קיימים



WinRAR

WinRAR היא תוכנה לקיבוץ קבצים ותיקיות. אולם, היא מציעה גם אפשרות לנעול את התיקיה/ הקבצים שנדחסו בעזרת הצפנת AES:



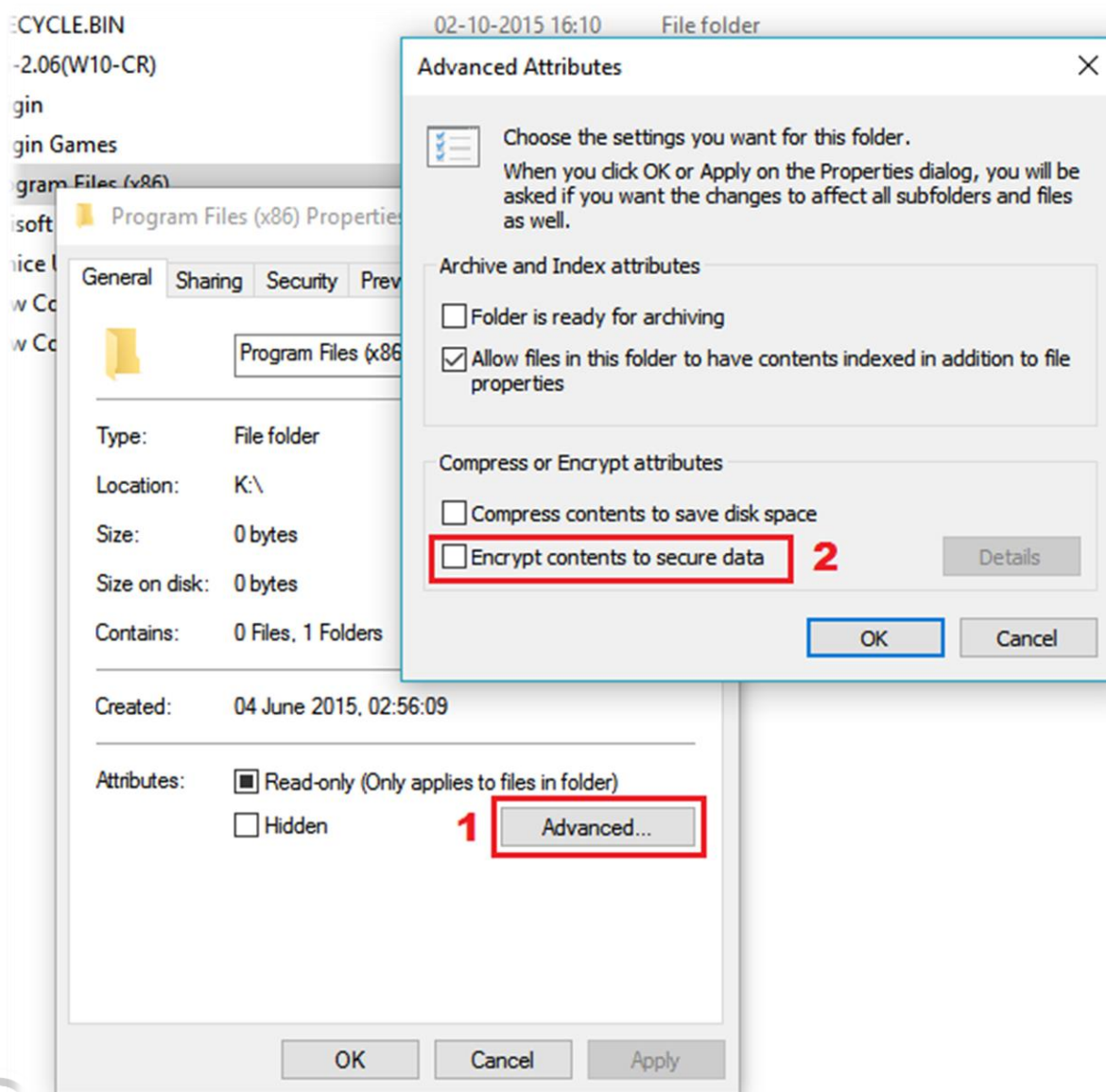
אולם, תכנית זו שונה מהמערכת שלי בכמה דברים:

- (1) מפתח ההצפנה והסיסמא נשמרים בתוך הקובץ הנעול/ מוצפן, ולא על שרת מרוחק, מה שעלול להוות פרצת אבטחה.
- (2) המטרה של מערכת WinRAR היא לדחוס קבצים, ולא לנעול אותם. לכן, ממשק המשתמש אינו קל ונוח לתפעול למי שרוצה לנעול את הקבצים בלבד.



Windows

מערכת ההפעלה windows מאפשרת להצפין קבצים כדי לנעול אותם.



גם כאן, חוזרים שני ההבדלים המרכזיים בין המערכת שלי לבין השירות שמציע windows:

- (1) מפתחות ההצפנה נשמרים בתוך הקובץ
- (2) כדי לנעול קובץ, יש להיכנס להגדרות הקובץ המתקדמות- ממשק מתקדם ומסובך, שעלול להקשות את התהליך עבור חלק מהמשתמשים.

מדריך ההתקנה למשתמש

צד השרת

יש להוריד את תיקיית השרת של הפרויקט, ולהריץ את הקובץ `server_main.py`. שימו לב: השרת דורש פייתון מותקן על המחשב כדי לרוץ.

צד הלקוח

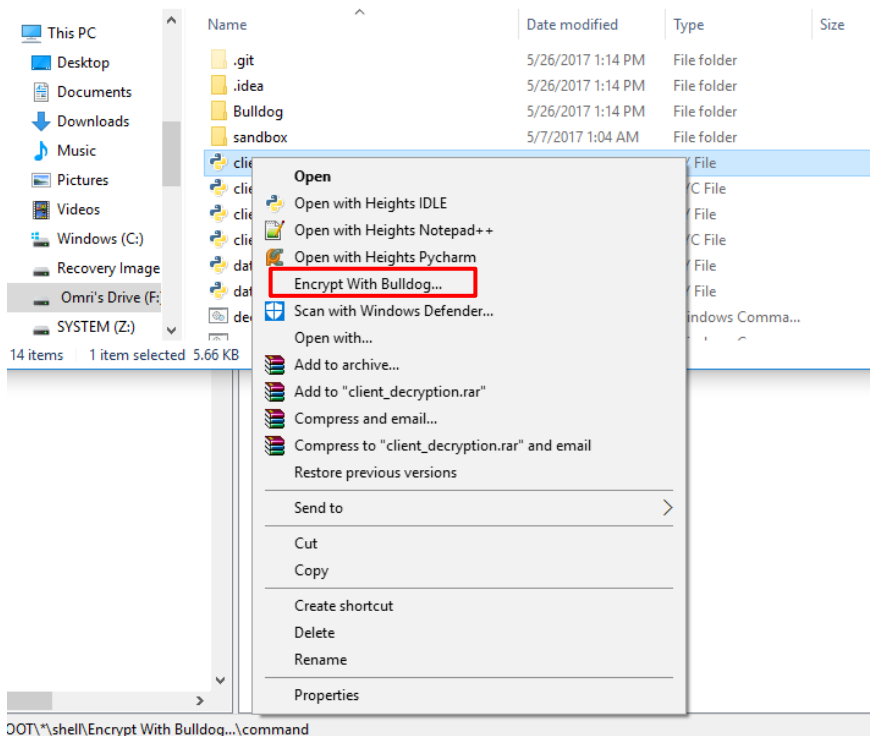
כדי להתקין את המערכת, יש להוריד את תיקיית הפרויקט, ולהריץ פעם אחת את הקובץ `setup.exe`. הקובץ יתקין את המערכת. לאחר מכן, ניתן להשתמש במערכת בקלות, דרך ה-context menu (התפריט שעולה בלחיצת קליק ימני). שימו לב: הלקוח אינו דורש פייתון מותקן על המחשב כדי לרוץ.



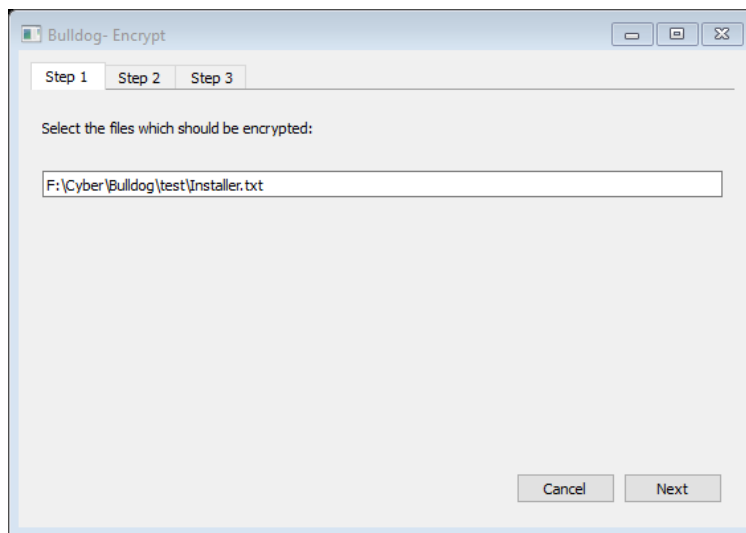
תיאור ממשק המשתמש

נעילת קובץ - הצפנה

תחילה, עלינו לסמן את הקובץ\התיקיה הרצויים ולבחור באפשרות ההצפנה:

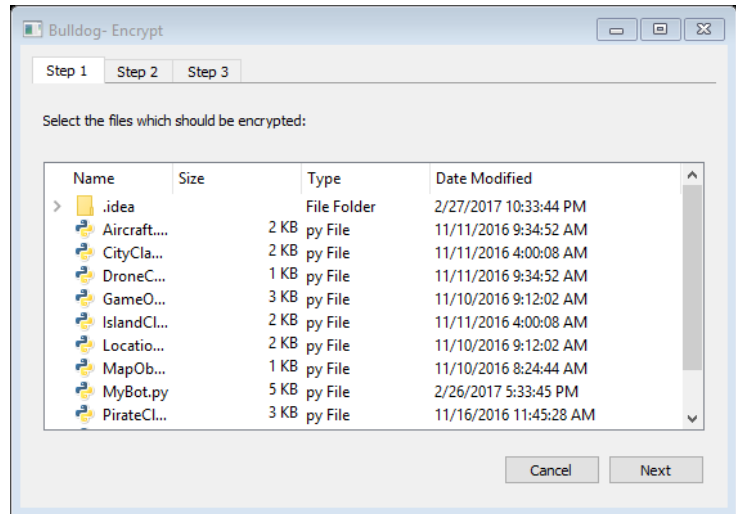


לאחר מכן, נקבל את החלון הראשון:

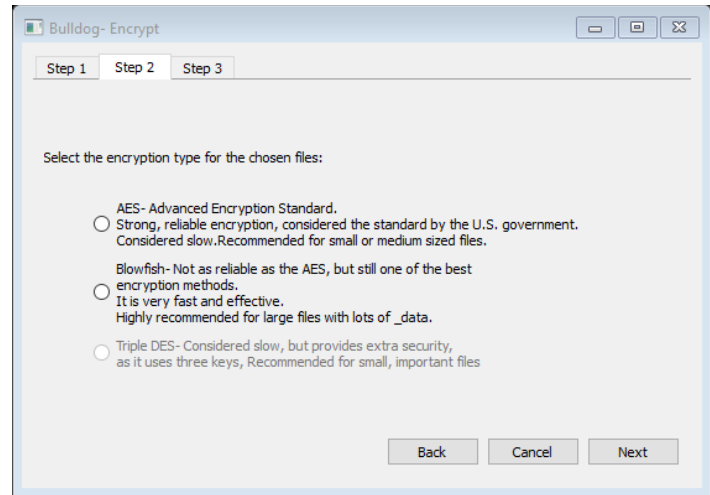


בחלון זה יוצג הקובץ בו בחרנו, או למקרה שבחרנו בתיקיה, יוצגו הקבצים שהתיקיה מכילה:

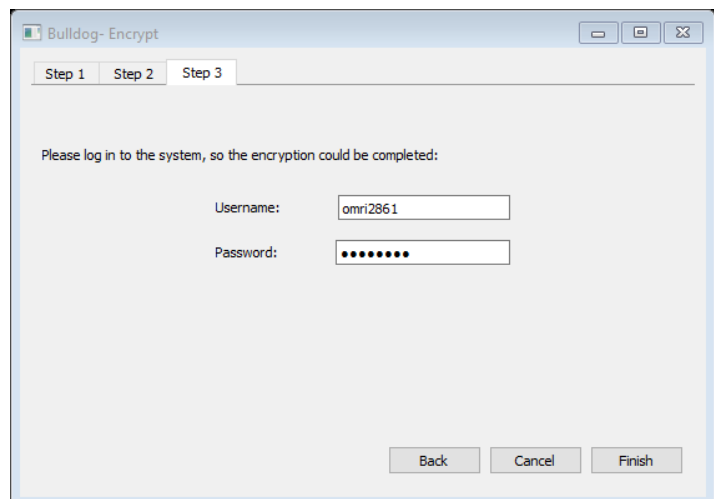




לאחר מכן, נצטרך לבחור בסוג ההצפנה הרצוי. נראה כי קיימים כמה סוגי הצפנה, כאשר לכל אחד מהם יש הסבר, המסביר לאילו צרכים הוא משמש:

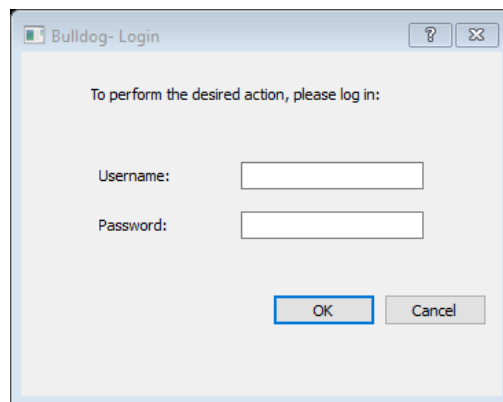


לאחר שבחרנו בסוג ההצפנה הרצוי, עלינו להתחבר למערכת:

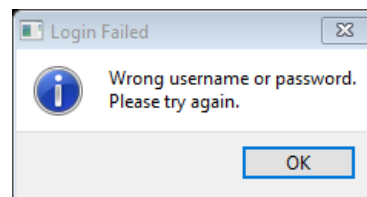


פתיחת קובץ - פענוח

לאחר שבחרנו בקובץ\ קבצים שברצוננו לפענח, או שננסה לפתוח אותו, Bulldog™ יפעל, והמערכת תנתח את הבחירה. אם הקובץ אינו קובץ שננעל על ידי Bulldog™, או שהתיקיה מכילה קבצים שאינם הוצפנו על ידי Bulldog™, או שבתיקיה מספר קבצים שננעלו על ידי משתמשים שונים, תתקבל הודעת שגיאה. במקרה שלא תתקבל הודעת שגיאה, יעלה החלון הבא, ונצטרך להתחבר למערכת:

A Windows-style dialog box titled "Bulldog- Login". It contains the text "To perform the desired action, please log in:". Below this are two input fields: "Username:" and "Password:". At the bottom right are two buttons: "OK" and "Cancel".

במידה ונתחבר עם המשתמש שנעל את הקבצים שנבחרו, הקבצים יפוענחו ויהיו פתוחים לגישה. במידה ולא נצליח, נקבל את הודעת השגיאה הבאה:

A Windows-style dialog box titled "Login Failed". It features an information icon (a lowercase 'i' in a blue circle) on the left. To the right of the icon is the text "Wrong username or password. Please try again.". At the bottom right is an "OK" button.

מבט אישי על העבודה ועל תהליך

הפיתוח

במהלך העבודה על הפרויקט, למדתי הרבה דברים חדשים וביצעתי מחקר מעמיק, בעיקר בתחום אבטחת המידע.

את תהליך העבודה חילקתי לכמה שלבים:

- **שלב ראשון- הגיית הפרויקט וכתובת מסמך האפיון-** חשבתי על רעיון הפרויקט הראשוני, יחד עם המורה ועוזר ההוראה. כתבתי את מסמך האפיון בהתאם לדרישותיהם. קיבלתי החלטות חשובות הקשורות לאופי הפרויקט: המטרות, הדרישות, הפיצ'רים, אופן הפיתוח (בצורה כללית), המטרה הסופית.
- **כתיבת מסמך העיצוב-** יצרתי תיאור כללי של ממשק המשתמש, כדי להבין בצורה טובה יותר את השימוש במערכת, את הפעולות שהמשתמש יבצע ובכך את אופן החלוקה לפונקציות בתכנית. לאחר מכן, יצרתי תרשים המתאר את רכיבי המערכת השונים ואת תפקידם באופן מפורט. הוא עזר לי להבין את מבנה המערכת, ולקבוע את הסדר שבו אני רוצה לפתח את הפרויקט.
- **מחקר בנושא ההצפנה-** ידעתי שאני רוצה לשלב מגוון סוגי הצפנה, ורציתי לדעת באילו סוגים של הצפנה אשתמש בדיוק, כחלק ממסמך העיצוב. לצורך כך, חקרתי הרבה על נושא ההצפנה, וקבעתי באילו סוגים של הצפנה אני מתכוון להשתמש בפרויקט שלי בדיוק. את ממצאי ניתן לראות תחת 'הגדרות' במסמך העיצוב (להוסיף תמונות מויקיפדיה ומאמרים).
- **כתיבת לוח הזמנים-** לאחר שהכרתי את מבנה המערכת והדרישות, והחלטתי לאיזה כיוון לקחת את הפרויקט, כתבתי את לוח הזמנים בהתאם לדרישות המורה ומשרד החינוך. לוח הזמנים כתוב בצורה מסודרת ומאורגנת, בטבלה המפרטת אילו חלקים יש לפתח, מתי, ומה יש לכלול בתוכם. לאחר מכן, התחלתי בפיתוח.
- **חלק ראשון בפיתוח- ההצפנה-** החלק הראשון בפיתוח, בהתאם ללוח הזמנים שקבעתי, הוא מודול ההצפנה. בתחילת העבודה חקרתי עוד בנושא ההצפנה- כיצד לשלב את סוגי ההצפנה שבחרתי בתוך הפרויקט. לאחר שמצאתי כיצד לבצע את ההצפנה, התחלתי בפיתוח. כתבתי מודול הכולל בתוכו מחלקה המבצעת את הצפנת הקבצים (הצפנה ופענוח), ושתי פעולות מרכזיות המצמצמות את כל הפעולות והעצמים במחלקה לנעילה\ פתיחה של קבצים. הקפדתי על תיעוד מלא בקוד, כתיבה מאורגנת ומודולרית, שתאפשר לי לבצע שינויים בהמשך במידת הצורך. מודול ההצפנה מסוגל להצפין ולפענח קבצים ב-3 שיטות שונות בצורה נוחה.
- **חלק שני בפיתוח- ממשק המשתמש-** בשלב זה, פיתחתי את מודול ה GUI כמעט בשלמותו, כדי שימחיש בצורה מלאה את כל דרישות המערכת, בהתאם לפעולות שיבצע המשתמש. תחילה, קראתי וחקרתי על מודולים קיימים שונים של GUI בפייתון, כדי למצוא את המודול המתאים ביותר לצורך שלי. מצאתי כי PyQt4 הוא המודול שישיר את מטרותיי בצורה הכי טובה. לאחר מכן, התחלתי בפיתוח ה GUI בעזרת כלי ה-designer הנוח של PyQt4. שילבתי את פעולות ההצפנה הקיימות, ולאחר שמימשתי רכיב כלשהו בפרויקט, מיד שילבתי אותו בממשק הגרפי. שיטת עבודה זו עזרה לי לעקוב אחר ההתקדמות שלי בצורה הטובה ביותר. מודול ה GUI היה כתוב בצורה מודולרית ונוחה, כך שהצלחתי לבצע בו שינויים בקלות.

בנוסף, המודול עושה שימוש באלמנטים גרפיים מתקדמים כמו event ו signal, כדי לוודא שה GUI מגיב באופן רציף ועובד בצורה יעילה.

- **חלק שלישי בפיתוח- השרת ומודול התקשורת-** בשלב זה פיתחתי את השרת, ללא מסד נתונים. מימשתי שרת multi-client, שיועד להשתמש בפרוטוקול ולקיים תקשורת מוצפנת. השרת ענה לבקשות הלקוחות בהתאם למידע שהיה כתוב אצלו בקוד, ולא במסד נתונים. המידע לא היה משמעותי באמת, זאת רק במטרה לממש את השרת. לאחר מכן, הטמעת גם את השימוש בפרוטוקול ובתקשורת מוצפנת בלקוחות, ובדקתי את אופן הריצה שלהם עם המידע הכתוב מראש.
- **חלק רביעי בפיתוח- מסד הנתונים-** בחלק הזה, למדתי SQL, והקמתי בסיס נתונים של SQL באמצעות python. לאחר מכן, חיברתי אותו לשרת. בשלב זה, הפרויקט היה כמעט מוכן (code freeze).
- **חלק חמישי בפיתוח- Bug Freeze-** תיקנתי את הבאגים האחרונים במערכת, והוספתי פיצ'רים קטנים לפרויקט, כדי ליצור מראה מקצועי ואיכותי יותר וליצור חווית משתמש אופטימלית.

לסיכום, נהייתי במיוחד לכתוב את השימוש ברשת- העמקתי במחלקות בפיתוח, וגם בשימוש ברשת, למדתי לעומק את אובייקט socket והתלבשתי עליו. זה היה חלק מעניין מאוד בפרויקט. נהייתי מאוד גם ללמוד SQL ולממש את בסיס הנתונים. זה כלי שאשתמש בו רבות בהמשך.

למדתי איך להתמודד עם קשיים בפרויקט, למשל, איך להתמודד עם בעיה שמונעת את המשך העבודה. בנוסף, למדתי גם איך לעקוב אחרי גרסאות בצורה ברורה ומובנת בעזרת git.



ביבליוגרפיה

שימוש בממש הגרפי

SourceForge PyQt4 Class Reference - תיעוד מלא עבור כל המחלקות של המודול הגרפי PyQt4:
<http://pyqt.sourceforge.net/Docs/PyQt4/classes.html>

התיעוד של Qt:

<http://doc.qt.io/qt-4.8/qtgui-module.html>

שימושים ב-slots and signals:

http://pyqt.sourceforge.net/Docs/PyQt4/new_style_signals_slots.html

https://www.tutorialspoint.com/pyqt/pyqt_signals_and_slots.htm

<http://zetcode.com/gui/pyqt4/eventsandsignals>

שימוש בהצפנה

האתר של הצפנת AES:

<http://aesencryption.net>

הערך של הצפנת AES בויקיפדיה:

https://en.wikipedia.org/wiki/Advanced_Encryption_Standard

הערך של TDES בויקיפדיה:

https://en.wikipedia.org/wiki/Triple_DES

הערך של Blowfish בויקיפדיה:

[https://en.wikipedia.org/wiki/Blowfish_\(cipher\)](https://en.wikipedia.org/wiki/Blowfish_(cipher))

איך להצפין עם Blowfish:

<https://stackoverflow.com/questions/35042164/how-to-decrypt-using-blowfish-in-pycrypto>

איך להצפין עם AES:

<https://stackoverflow.com/questions/20852664/python-pycrypto-encrypt-decrypt-text-files-with-aes>

איך להצפין עם RSA:

<https://stackoverflow.com/questions/30056762/rsa-encryption-and-decryption-in-python>



נושאים נוספים

Python multiprocessing

<https://docs.python.org/2/library/multiprocessing.html>

Python threading

<https://docs.python.org/2/library/threading.html>

Python struct

<https://docs.python.org/2/library/struct.html>

Python os module

<https://docs.python.org/2/library/os.html>

שימוש במודול py2exe:

<http://www.py2exe.org/index.cgi/Tutorial>

