



שם בית הספר: תיכון חדש הרצליה

שם העבודה: VPN (Virtual Private Network)

שם התלמיד: אריאל גלזמן

ת.ז התלמיד: 213497373

שם המנחה: מיכאל צ'רנובילסקי

שם החלופה: "הגנת סייבר ומערכות הפעלה"

תאריך הגשה: 24.5.2021

## תוכן עניינים

3	מבוא
3	- תיאור תכולת הספר
3	- הרקע לפרויקט
3	- תהליך המחקר
3	- אתגרים מרכזיים
4	מבנה / ארכיטקטורה של הפרויקט
4	- תרשים <i>Top-down levels diagram</i>
5	- תרשים <i>Top-down levels diagram</i> לזרם מידע בין מערכות
6	מבנה נתונים
7	יחסי שרת-לקוח/ארכיטקטורת רשת
7	- תרשים
8	- מבנה הודעה
9	- התקן <i>TUN\TAP</i>
10	- ניתוב חבילות על פי טבלת הניתוב
11	- ביצוע שינויים על טבלת הניתוב
11	- בעיית ה- <i>super user</i>
13	תיאור הצפנות
13	- הצפנה והכנות אליה
14	- העברת המפתח הסימטרי באמצעות <i>RSA</i>
15	תיאור האלגוריתם הראשי
15	- תרשים
17	- שינויים על החבילה מהיציאה ועד חזרה
19	<i>Activity Diagram</i>
19	- תרשים מבנה השרת
20	- תרשים מבנה הלקוח
22	<i>Design Classes Diagram</i>
22	- תרשים מבנה השרת
23	- תרשים מבנה הלקוח
24	טכנולוגיות בהן נעשה שימוש
24	API-ים
25	מדריך למשתמש
26	מדריך למפתח
43	רפלקציה אישית
44	ביבליוגרפיה

## מבוא

תיאור תכולת הספר:

לקורא שלום רב, הספר הינו תיעוד של המערכת שהינה פרויקט הגמר שלי, בקצרה אומר שהמערכת מצפינה ומנתבת את תעבורת הרשת של משתמש, המכוונת לצד שלישי, למחשב מרוחק על מנת שזה יבצע את התקשורת עם הצד השלישי במקום המשתמש. להלן הסבר על עיקרי הספר, פרק המבוא,

הרקע לפרויקט:

הפרויקט הינו ביטוי של צורך באבטחה של מידע, אני באופן אישי חרד מאוד לאבטחת המידע שלי, ופרויקט כזה התאים לי ככפפה, בנוסף בפרויקט יש שימוש רב בנושאים של רשתות בכלים של UNIX ובסביבת לינוקס שהיו חדשים ולא מוכרים לי, ומצא חן בעיני הצורך ללמוד אותם ולהוסיף אותם לאמתחת היכולות האישיים.

תהליך המחקר:

תוכנות VPN הם דבר מאוד מוכר, משתמשים בהן למטרות שונות, בין המשתמשים פרטיים המבקשים להסתיר את תעבורת הרשת שלהם, או לזייף את מיקומם על מנת לקבל שירותים שאפשר לקבל במדינות אחרות, גם חברות משתמשות בזה בעיקר למטרות של אבטחת מידע והגבלת גישות, משתמשים בזה לפעמים גם אקטיביסטים במדינות טוטליטריות המגבילות כניסה לשרתים שמסכנים את שלטונן או מצותות לאזרחיה. תוכנות מוכרות בשוק הן:

- AVG VPN
- TunnelBear
- NordVPN

אתגרים מרכזיים:

אחד האתגרים המרכזיים שאיתם התמודדתי הוא ההתרגלות לסביבת לינוקס ובניית סביבת העבודה, בהמשך הספר מוצגים הסברים לבעיות ולפתרונות לבעיות הללו.

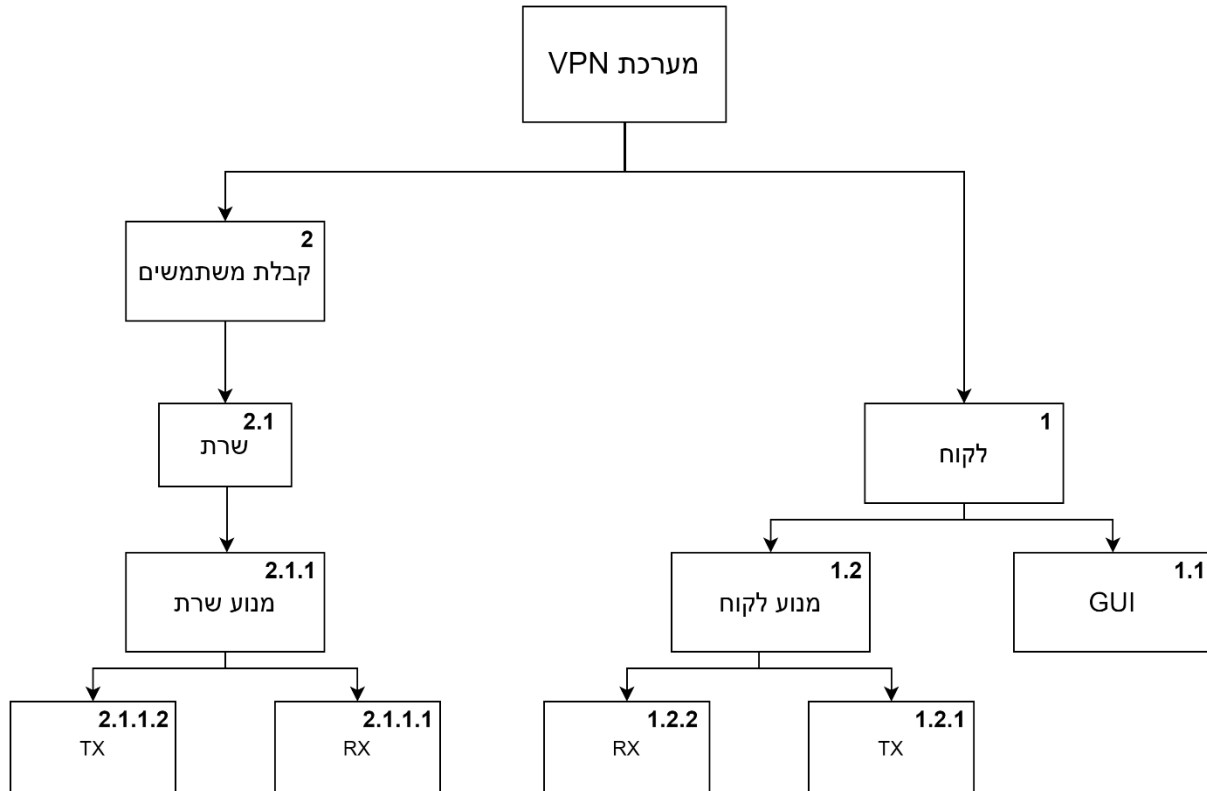
הצגת פתרונות לבעיה:

הבעיה שהפרויקט בא לפתור הוא הצורך בהסתרת המידע וזהות מקור המידע ובמידת הצורך, גם מיקום לוגי או גיאוגרפי. בעייה זו איננה חדשה, וקיימים לה פתרונות רבים – אך בפרויקט הבאתי לידי ביטוי את היכולות של שינויים במסגרת מערכות שרת-לקוח, אשר משולבים ביכולות של מחשב ביתי המריץ בעיקר ווינדוס WINDOWS עם שימוש נרחב באמולציות של מערכת הפעלה לינוקס – LINUX ע"י שימוש ב-DISTRIBUTION / הפצה של UBUNTU המשמש ככלי לשימוש וביצוע מניפולציות של רמות שונות ברשת ה-IP.

הסתרת המידע הינה נדבך חשוב על מנת לבצע הצפנה נדרשת ידיעה של מפתח הצפנה בין שני הצדדים. לשם כך, נעשה שימוש בפרוטוקול ההצפנה האסימטרית שלה יתרונות רבים ונפוצה מאד בשימוש באינטרנט היום. דוגמאות לכך הן: TLS המשמש לרכישה וביצוע פעולות רגישות ברשת.

# מבנה / ארכיטקטורה של הפרויקט

תרשים Top-down levels diagram:



## 1 – לקוח

1.1 – ממשק המשתמש (GUI) מפעיל ומכבה את מנוע הלקוח

1.2 – מנוע הלקוח הינו החלק הפעיל של הלקוח שאחראי לשינויים במערכת ולתקשורת עם השרת

1.2.1 – מנגנון האחראי לקרוא חבילות מהלקוח ולשלוח אותן אל השרת

1.2.2 – מנגנון האחראי לקרוא חבילות מהשרת ולכתוב אותן אצל הלקוח

2 – קבלת המשתמשים הינו תהליך נפרד שאחראי להסדיר תקשורת עם הלקוח ומפריט אותו

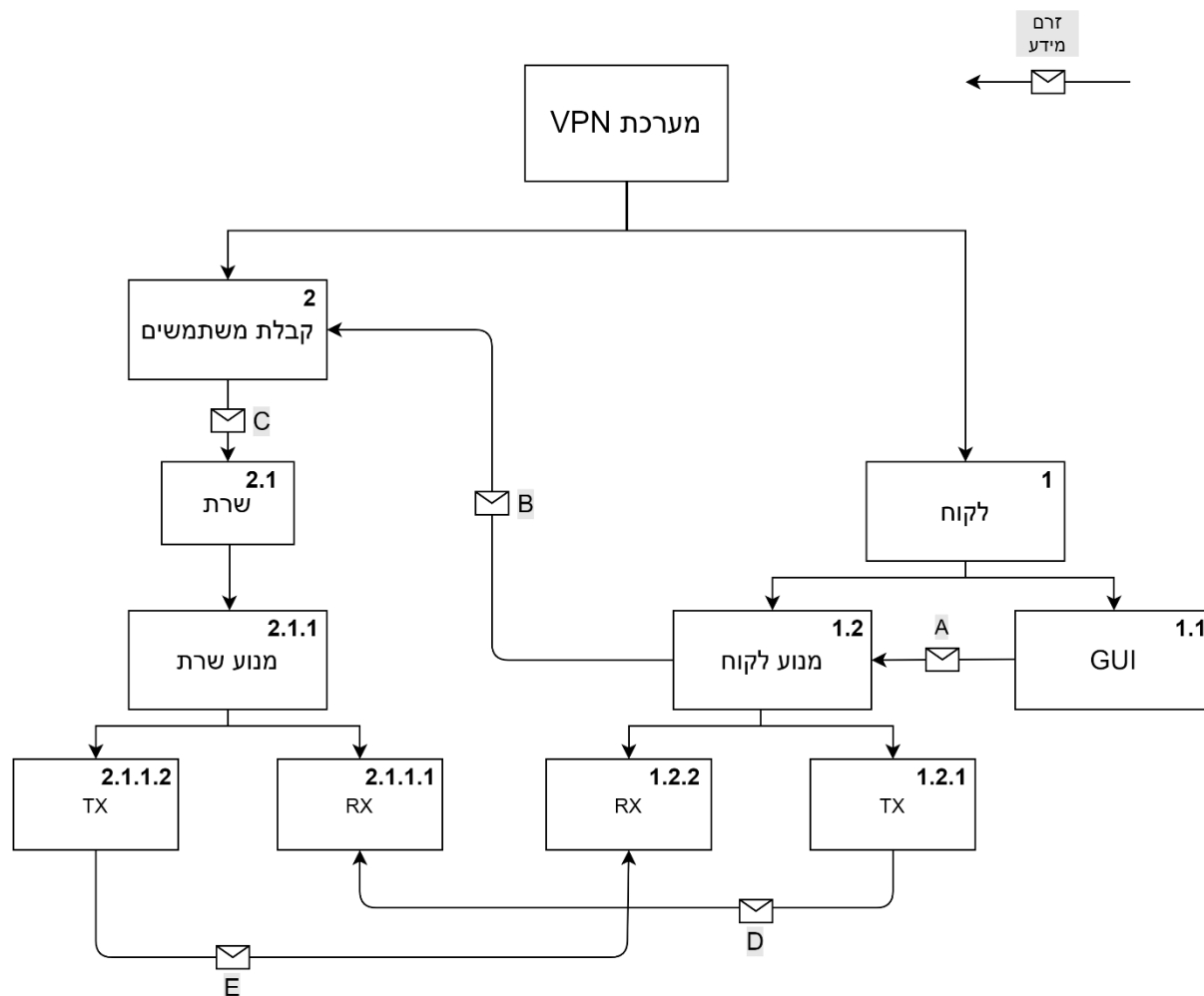
## 2.1 – שרת

2.1.1 – מנוע השרת אחראי על התקשורת מול הלקוח ומול היעד, על הצפנה ועל שינויים במערכת.

2.1.1.1 – מנגנון האחראי לקרוא מהלקוח חבילות ולכתוב אותן החוצה אל היעד

2.1.1.2 – מנגנון האחראי לקרוא חבילות מהיעד ולשלוח אותן אל הלקוח

תרשים Top-down levels diagram לזרם מידע בין מערכות:



A - אות להתחלה וסיום פעילות

B - מפתח סימטרי (32 בטים)

C - פרטי משתמש ו-Socket

D - חבילה שנקראה אצל הלקוח ונשלחת אל השרת ע"מ שתגיע ליעד

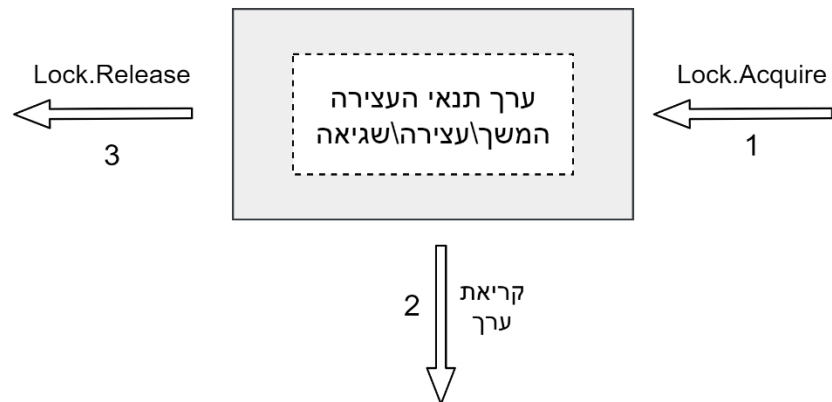
E - חבילה שנקראת מהיעד אצל השרת ונשלחת אל הלקוח

## מבנה נתונים:

ישנו "מבנה נתונים" אחד חשוב במערכת אשר נמצא אצל הלקוח ואצל השרת, הערך הוא תנאי העצירה. תנאי העצירה נמצא בזיכרון משותף לכל התהליכים, והוא נגיש לכמה וכמה תהליכים.

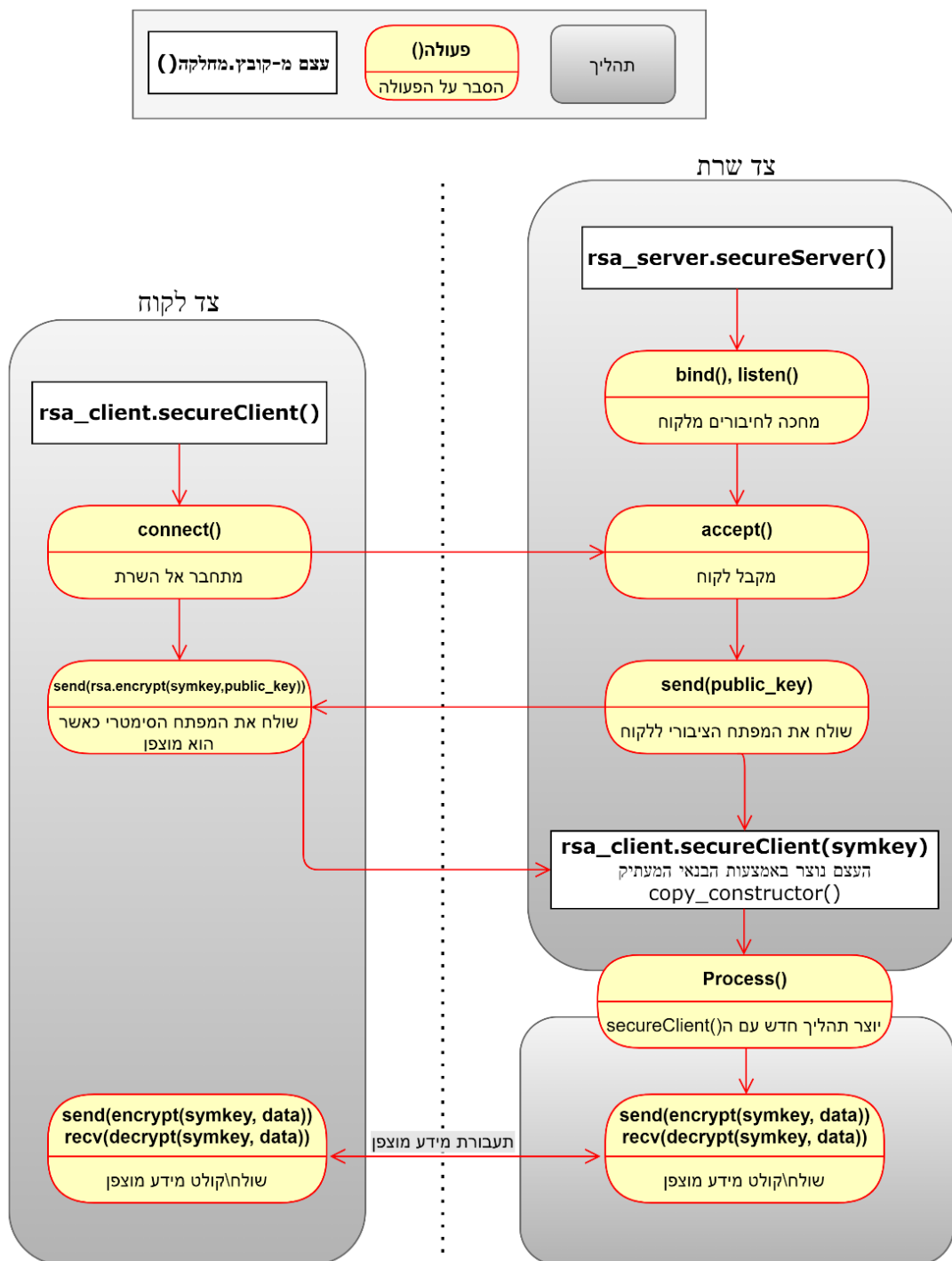
התהליכים לא מסונכרנים עם הפניות שלהם, והם באופן קבוע דורשים את תנאי העצירה כדי לדעת אם להמשיך או להפסיק. גישה סימולטנית עלולה לשבש את תנאי העצירה, להביא ערך שגוי או למנוע מערך להתעדכן.

לכן יש שימוש ב<sup>1</sup>Lock על מנת למנוע גישה בו-זמנית אל תנאי העצירה.



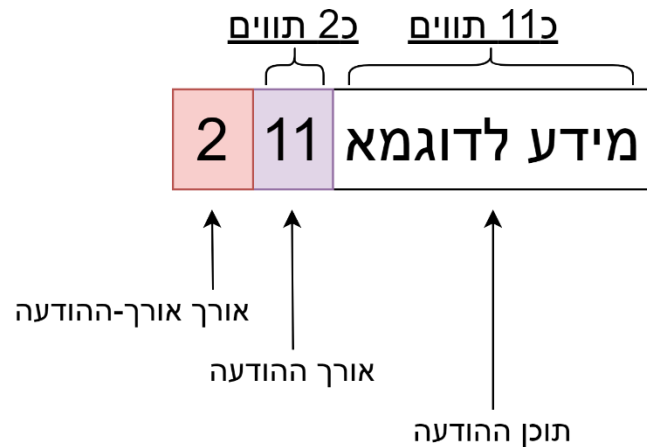
## יחסי שרת-לקוח/ארכיטקטורת רשת:

תרשים:



השרת מחכה לחיבור, מתחבר, שולח את המפתח הציבורי אל הלקוח, הלקוח מייצר את המפתח הסימטרי מצפין אותו עם המפתח הציבורי ושולח לשרת, השרת מפענח באמצעות המפתח הפרטי את ההודעה ומשיג את המפתח הסימטרי, השרת מייצר `secureClient()` באמצעות הבנאי המעתיק שלו `copy_constructor()` ושולח אותו אל תהליך חדש ובזאת מוותר עליו בעצם ומתפנה לקבלה של לקוחות חדשים, התהליך הנוסף הינו עכשיו השרת אל מול הלקוח, מבחינת מחלקות/עצמים שניהם `secureClient`, אחד אשר בשרת ואחד בלקוח.

מבנה הודעה:



מבנה ההודעה בנוי מ-3 רכיבים הרכיב בקצה ההודעה הוא תוכנה, המסר שרוצים להעביר. מאחוריו בסדר אורכה של ההודעה במספר. מאחוריו הוא האורך של אורכה של ההודעה.

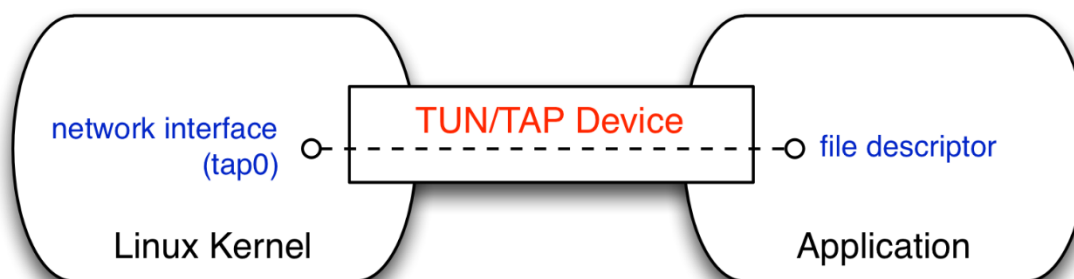
בדוגמא למעלה, תוכן ההודעה מכיל 11 תווים, לכן אורך ההודעה נרשם, 11. אורכו של אורך ההודעה הוא 2 תווים ולכן נרשם בראש ההודעה 2.

צד הקורא את ההודעה יודע כי עליו לקרוא דבר ראשון תו אחד שהוא מספר התווים ברכיב אורך ההודעה, על פי זה, דרך אגב הוקצה בעצם תוכן ההודעה לאורך 9 ספרתי, כלומר עד 999,999,999 בתים (בלי שני הרכיבים הנוספים, שהם זניחים).

לאחר שקרא את מספר הספרות של אורך ההודעה, הוא קורא את אורך ההודעה, ולאחר מכן את תוכנה של ההודעה. לאחר מכן קורה תהליך הפענוח.



## התקן TAP\TUN<sup>2</sup>:

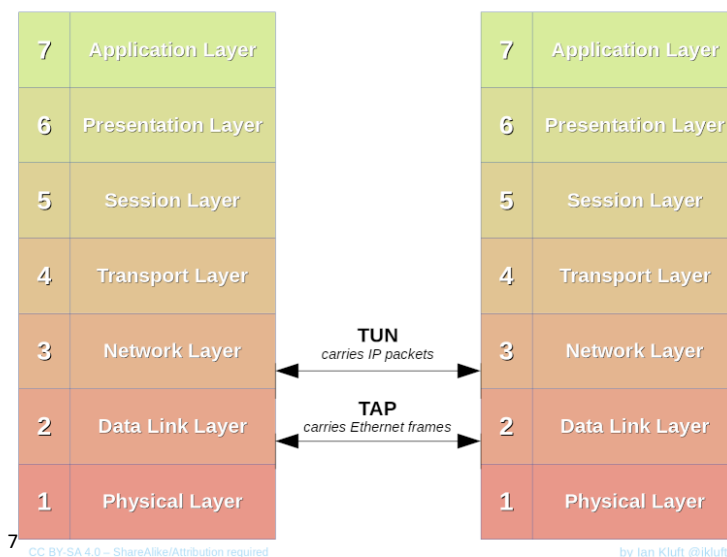


התקן TAP\TUN הוא התקן NIC<sup>3</sup> (Network Interface Controller) וירטואלי, ההתקן כולו תוכנה ולא נמצא פיזית במחשב. להתקן הגדרות כמו להתקנים הפיזיים, ואפשר לערוך הגדרותיו באמצעות ספריית הפעולות (ifconfig<sup>4</sup>), להפעילו, לכבותו, להגדיר לו כתובות, לקבל עליו מידע וכו'. ההתקן נפתח באמצעות פתיחת (os.open()<sup>5</sup>) הקובץ "dev/net/tun" לקריאה ולכתיבה (os.O\_RDWR). להתקן הגדרות נוספות המגדירות את אופן הפעולה שלו<sup>6</sup>,

IFF\_TUN – קובע כי התעבורה שיקלוט ההתקן תהיה מהשכבה ה-3 של הרשת

IFF\_TAP – קובע כי התעבורה שיקלוט ההתקן תהיה מהשכבה ה-2 של הרשת

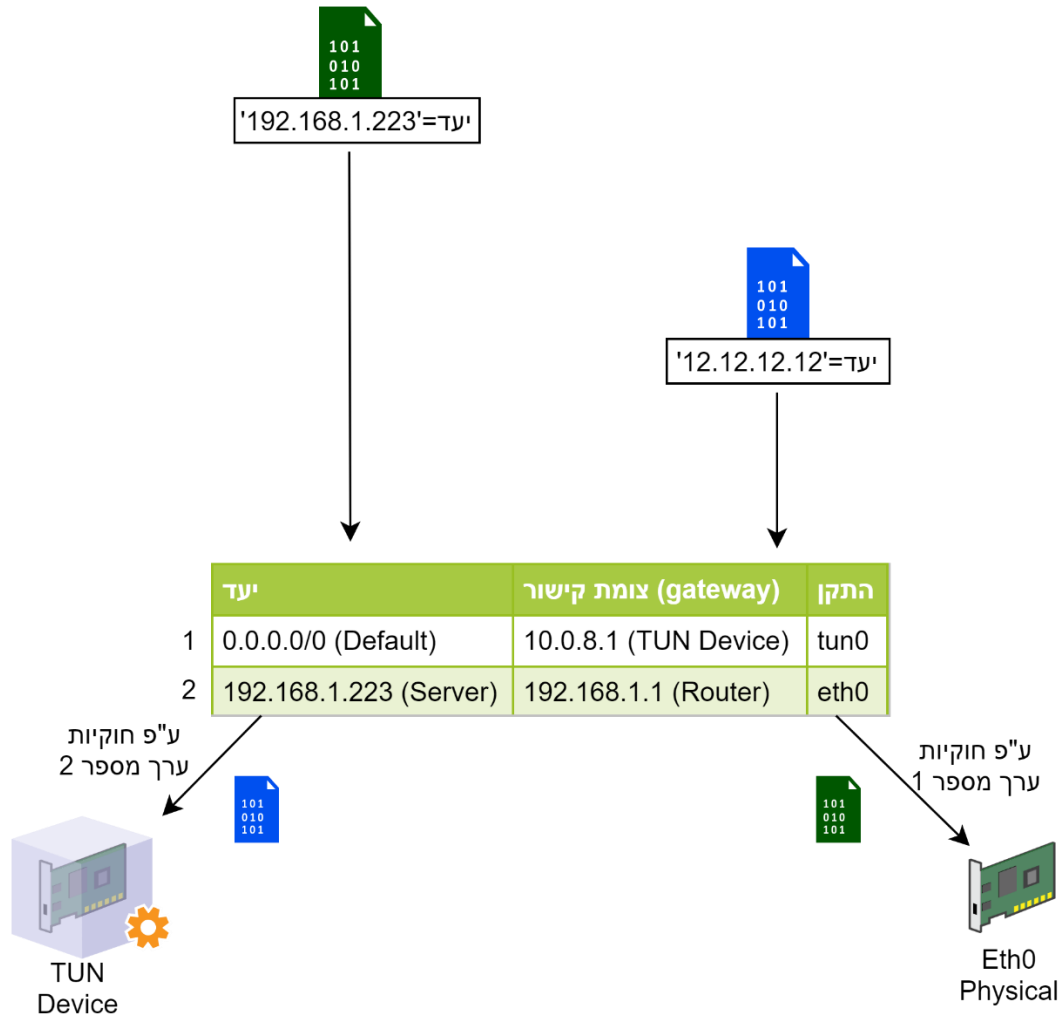
## TUN and TAP in the network stack



IFF\_NO\_PI – קובע כי התעבורה לא תכלול מידע כמו דגלים וכו'.

על מנת לערוך שינויים על ההתקן החדש משתמשים בספריית ioctl שהיא ספריית C הניתנת לשימוש בפייתון<sup>8</sup>. ספרייה זו עוזרת לערוך את הגדרות ההתקן ושמו. השם שישלח אל המערכת יכול להיות קבוע או בצורת Format-String<sup>9</sup>, בו המערכת תקבע את השם עם הערך הפנוי הבא איפה שנמצא ב-Format-String. למשל "tun\_dev%d" יקבל "tun\_dev0" ולאחר מכן "tun\_dev1" וכו'.

ניתוב חבילות ע"פ טבלת הניתוב<sup>10</sup>:



טבלת הניתוב היא הרכיב במערכת ההפעלה שאחראי לקבלת ההחלטות לניתוב חבילות אל התקני הרשת, החבילה תנתב לכתובת הקרובה ביותר אליה על-פי החוקים בטבלה. על מנת לערוך שינויים על החבילה, להצפין אותה ולהטמין אותה בחבילה אחרת, צריך לנתב אותה אל התקן הTUN. כדי שתנתב החבילה צריך לשנות את החוקים של טבלת הניתוב.

התוכנה מבצעת שינויים בטבלת הניתוב בהפעלת התוכנה "START" ומחזירה את החוקים למצבם הראשוני בכיבויה "STOP". בדוגמה לעיל מוצגת טבלת הניתוב **כאשר התוכנה מופעלת**.

בדוגמה ישנן שתי חבילות המגיעות אל סף הניתוב, האחת לכתובת היעד '12.12.12.12' והשנייה אל כתובת היעד '192.168.1.223', החבילה הראשונה ('12.12...') מתאימה לחוק 1, ואיננה מתאימה לחוק 2 ולכן תנתב על פיו להתקן tun0, החבילה השנייה ('...1.223') מתאימה לחוק 1 וגם לחוק 2, חוק 2 הינו המדויק ביותר ולכן החבילה תנתב על פיו אל ההתקן Eth0.

## ביצוע שינויים על טבלת הניתוב:

הבה נביט בטבלת הניתוב בברירת המחדל שלה:

```
ubuntu1@ubuntu1-VirtualBox:~$ route
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
default OpenWrt.lan 0.0.0.0 UG 0 0 0 enp0s3
10.0.8.0 0.0.0.0 255.255.255.0 U 0 0 0 tun_dev1
link-local 0.0.0.0 255.255.0.0 U 1000 0 0 enp0s3
192.168.1.0 0.0.0.0 255.255.255.0 U 100 0 0 enp0s3
ubuntu1@ubuntu1-VirtualBox:~$
```

אנו רואים שישנו מכשיר אחד שכל תעבורה מכל כתובת אשר תגיע אליו, ולשכבה הנוספת לא מנותב דבר.

נוסיף כאן מימד נוסף, מכשיר חדש – ניתן לו כתובת ונביט שוב בטבלת הניתוב – מאחר ולא ביצענו שום שינוי – מצופה שהתעבורה לא תינתב אליו. כעת נשתמש בפקודות הבאות:

- **sudo route add 0.0.0.0/0 gw 10.0.8.0 dev \*name\***
- **sudo route del 0.0.0.0/0 gw 192.168.1.1 dev \*eth\_device\***

בזאת הסטתי את הפקטות מדרכן מהמכשיר הפיזי אל הווירטואלי.

אמנם, כן צריכה להנתן גישה למחשב אחד דרך המכשיר הפיזי, וזה אל השרת, ולכן נדרש עוד ערך אחד חריג למחשב אחד.

- **sudo route add 192.168.1.x/32 gw 192.168.1.1 dev \*eth\_device\***

בזאת הסתיימה עבודת הניתוב על לעצירת התוכנה ובעייתה על כל ערך שנרשם del ירשם add ולהיפך.

כאשר החבילות מועברות להתקן TUN ניתן לקרוא אותן באמצעות `os.read()` הקוראת מהקובץ.

## בעיית ה-super user:

פעולות מהסוג שתיארתי לעיל דורשות הרשאות super-user בעברית "מנהל מערכת" ובשפת לינוקס `sudo`.


אחרת עולה הודעה אשר מסבירה כי אין ברשות פייתון לבצע את הפעולה:

**Error: Operation not permitted[1]**

הרצת סקריפט פייתון עם הרשאות `sudo` פשוטה למדי, כל אשר נדרש לעשות הוא לפתוח את המסוף ולכתוב `sudo python script_name.py` להכניס את הסיסמא והסקריפט רץ עם הרשאות מנהל מערכת, אבל העיסוק הזה בכל הרצה מטיל צל כבד על הפיתוח והדיבוג, מקשה על העבודה ומאט אותה, ורצוי לעבוד בסביבת העבודה הנוחה, אבל כיצד אגרום לה להריץ סקריפטים על `sudo`?

על מנת להריץ את הסקריפט בסביבת העבודה (PyCharm) נדרש להחליף את המפרשן (Interpreter) לבאש-סקריפט (Bash Script) אשר יריץ את הסקריפט עם הרשאת מנהל-מערכת, כלומר יבצע פקודת `sudo python3 script_name.py`, המפרשן הסטנדרטי, מקבל את שם הקובץ ברשימה של שמות שם הארגומנטים (האפשריים – לא חובה) שמזין המשתמש לדוגמא אם נרצה להריץ איזשהו סקריפט נרשום את הארגומנטים עם רווחים אחרי שנזין את שם הקובץ,

`root$ python3 script.py 1 + 1`



`sys.argv[0,1,2,3]`

רשימה זו מוכרת לנו כ-`sys.argv` ובמקום ה-0 בה נמצא שם הקובץ והארגומנטים הנוספים במקומות הבאים, נרצה לכתוב באש-סקריפט שיריץ את פייטון `sudo` שיקבל את הארגומנטים שהם שם הקובץ וכל ארגומנט נוסף, לכן סקריפט-מסוף שנרצה לצרף למפרשן יראה כך:

```
| python-sudo.sh |
1 sudo python3 "$@" # כל ארגומנט שיוזן
2
```

על מנת להפוך את הבאש-סקריפט לבר-הרצה<sup>11</sup> נשתמש בפקודה:

```
root$ sudo chmod +x python-sudo.sh
```

כאשר אנחנו נמצאים בתיקייה בה נמצא הקובץ

אבל, כאשר מריצים את הבאש-סקריפט מערכת ההפעלה עדיין דורשת מהמשתמש להזין סיסמא כשירצה להריץ דבר מה עם הרשאות SUDO, כיצד אפשר לעקוף זאת? על מנת שכמשתמש אוכל להשתמש בפקודות עם הרשאות `sudo` מבלי להזין בכל פעם סיסמא וזאת כדי שסביבת העבודה תוכל לרוץ עם הרשאות SUDO – יש לשנות קובץ אשר ממנו מערכת ההפעלה יונקת את החוקים המגדירים לאילו משתמשים יש את היכולת לעבוד כמנהל מערכת ללא הזנת סיסמא. הקובץ הזה נמצא במיקום `/etc/sudoers`, על מנת לערוך את הקובץ בצורה בטוחה ניתן להשתמש בכלי הנקרא `VISUDO`<sup>12</sup>, שימוש בכלי זה יעשה באמצעות הפקודה:

```
root$ sudo visudo
```

נוסיף את השורה הבאה בסוף הקובץ,

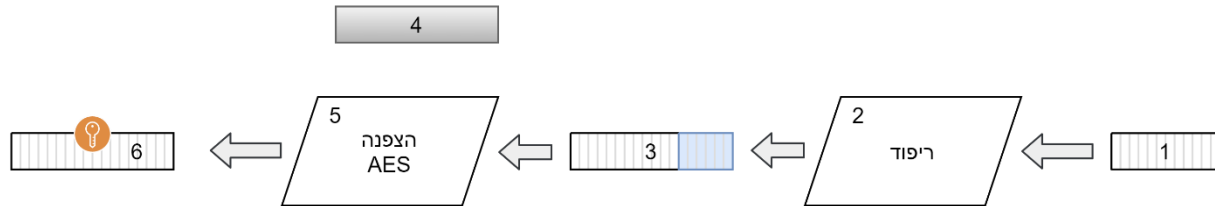
```
%ubuntu2 ALL=(ALL) NOPASSWD:ALL
```

במקרה זה, `ubuntu2` הוא שמו של המשתמש, מטבע הדברים יכול להיות שם שמו של כל משתמש אחר.

`Ctrl+V` כדי לשמור, `Ctrl+X` כדי לצאת. משלב זה כל פקודה או תוכנה אשר מריצה פקודה ממשתמש `ubuntu2` יכולה לבקש הרשאות SUDO מבלי להזין סיסמא.

## תיאור הצפנות:

הצפנה והכנות אליה:



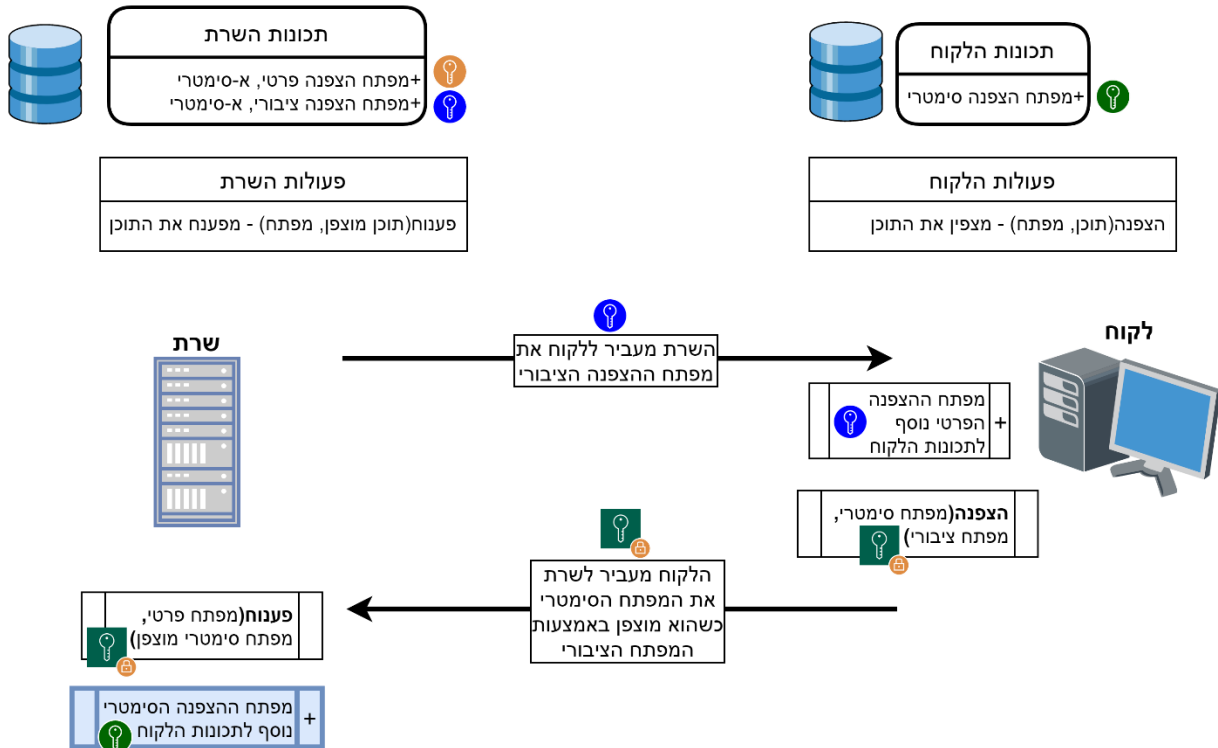
- 1 – מידע גלוי, בגודל כלשהו
- 2 – פעולת ריפוד משמעותה לחבר מידע סתמי בגודל נחוץ למידע הגלוי כדי להשלימו לגודל מפתח ההצפנה
- 3 – מידע מרופד בגודל מפתח ההצפנה
- 4 – מפתח ההצפנה
- 5 – הצפנת המידע המרופד במפתח ההצפנה
- 6 – מידע מוצפן

AES הוא צופן בלוקים סימטרי, צופן סימטרי הוא אלגוריתם הצפנה שבו משתמשים במפתח ההצפנה יחיד הן להצפנה של הטקסט הקריא והן לפענוח של הטקסט המוצפן.

אופן הפעלה של צופן בלוקים שהשתמשתי בו הוא ECB.  
תכונותיו של אופן ECB<sup>13</sup>:

- **ביטחון**; בלוקים של טקסט-גלוי זהים מפיקים תמיד בלוקים של טקסט-מוצפן זהים - כל עוד מפתח ההצפנה זהה. לכן קל לזהות שהוצפנו בלוקים זהים.
  - **כשל מצטבר**; שגיאה בסיבית אחת או יותר של בלוק טקסט-מוצפן משבשת את פענוח הבלוק הגלוי המתאים בלבד. יתר הבלוקים אינם משתבשים. פענוח של בלוק המכיל סיבית שגויה אחת יהיה אקראי, דהיינו בממוצע 50 אחוז מסיביות הבלוק יהיו שגויות.
  - **תלות הדדית**; הבלוקים מוצפנים בנפרד ללא תלות זה בזה. שינוי סדר הבלוקים המוצפנים גורר אחריו לאחר פענוח שינוי בסדר הבלוקים הגלויים בהתאם. סידור מחדש או הסרה זדונית של בלוקים שלמים עלולים שלא להתגלות.
- המידע מוצפן בבלוקים של מידע, הבלוקים של המידע לא בהכרח יגיעו לגודלו של מפתח ההצפנה ולכן יש להתאימן לגודלו, תהליך זה נקרא ריפוד (padding), והוא הזנה של מידע סתמי שיוצפן ויוסר לאחר שיפוענח.

## העברת המפתח הסימטרי באמצעות RSA:



על מנת לבצע את ההצפנה עם פרוטוקול AES, חייב להיות מפתח יחיד, סימטרי, אצל שני הצדדים. המפתח צריך לעבור באופן מוצפן.

פרוטוקול RSA מאפשר להעביר מידע בלי מפתח ידוע מראש, הוא פרוטוקול שצורך משאבי חישוב רבים ולכן משתמשים בו רק להעברת המפתח הסימטרי, כשיש לשני הצדדים את המפתח הסימטרי אפשר להעביר מידע מוצפן.

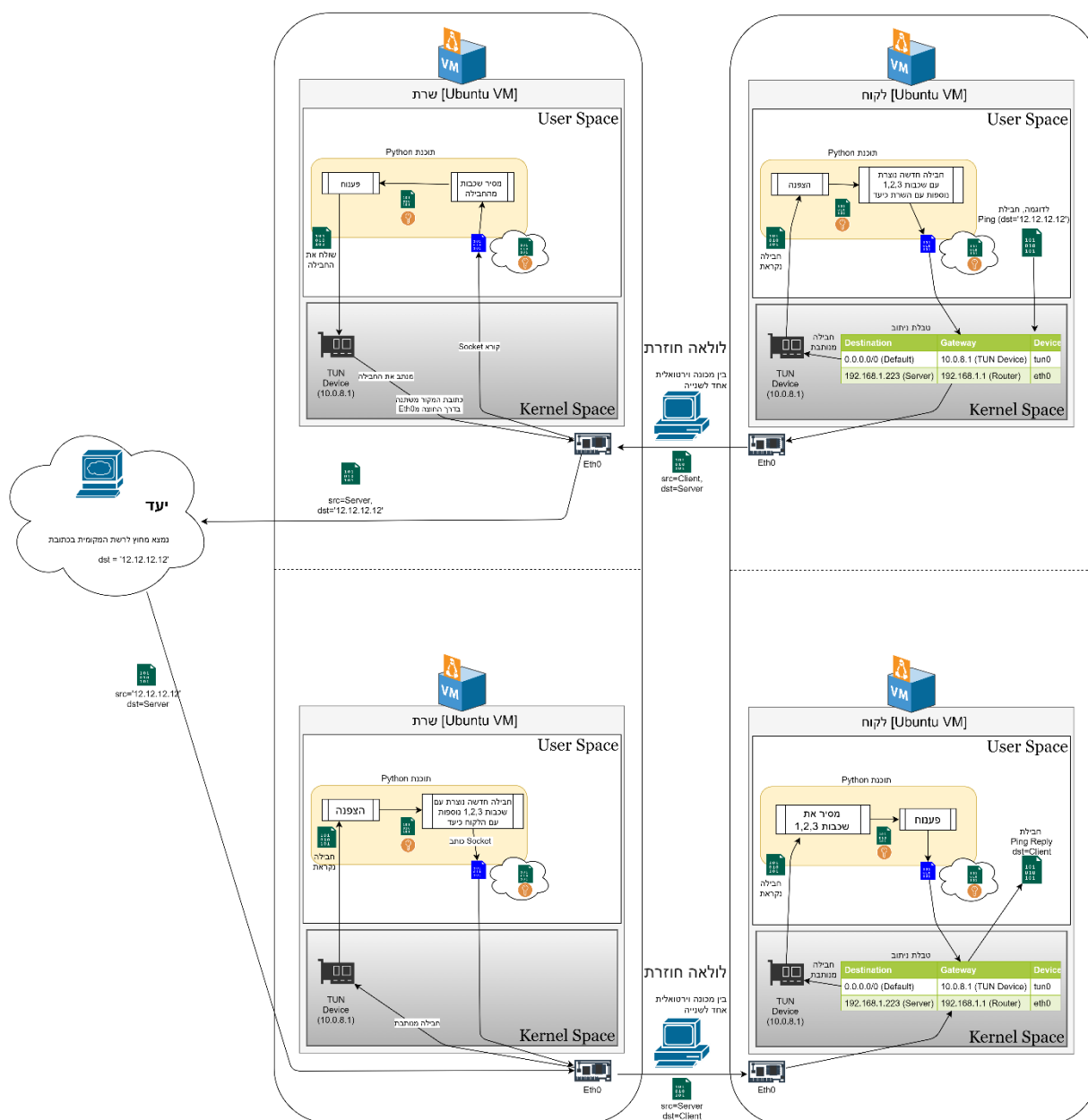
צד אחד מחזיק בשני מפתחות, עם מפתח אחד ניתן רק להצפין את המידע (והוא ציבורי) ועם השני ניתן רק לפענח את המידע (והוא פרטי). בתוכנה, השרת מייצר את שני המפתחות הא-סימטריים והלקוח מייצר מפתח סימטרי (באורך 32 בתיים).

השרת שולח את המפתח הציבורי ללקוח, הלקוח מצפין באמצעותו את המפתח הסימטרי, שולח לשרת והשרת מפענח את ההודעה עם המפתח הפרטי ומשיג את המפתח הסימטרי.

## תיאור האלגוריתם הראשי:

הלקוח מגיע ליעד דרך שרת, זהו האלגוריתם, חבילה מוצפנת אצל הלקוח מועברת אל השרת השרת מפענח אותה ושולח ליעד. חבילה שנשלחת מהיעד לשרת מוצפנת ומועברת ללקוח.

תרשים:



בתרשים ישנה חבילת Ping Request היוצאת מהלקוח ומנתבת אל התקן TUN, התוכנה קוראת ממנו את החבילה, מצפינה אותו ומוסיפה שכבות נוספות לחבילת TCP חדשה מהלקוח לשרת ושולחת אותו

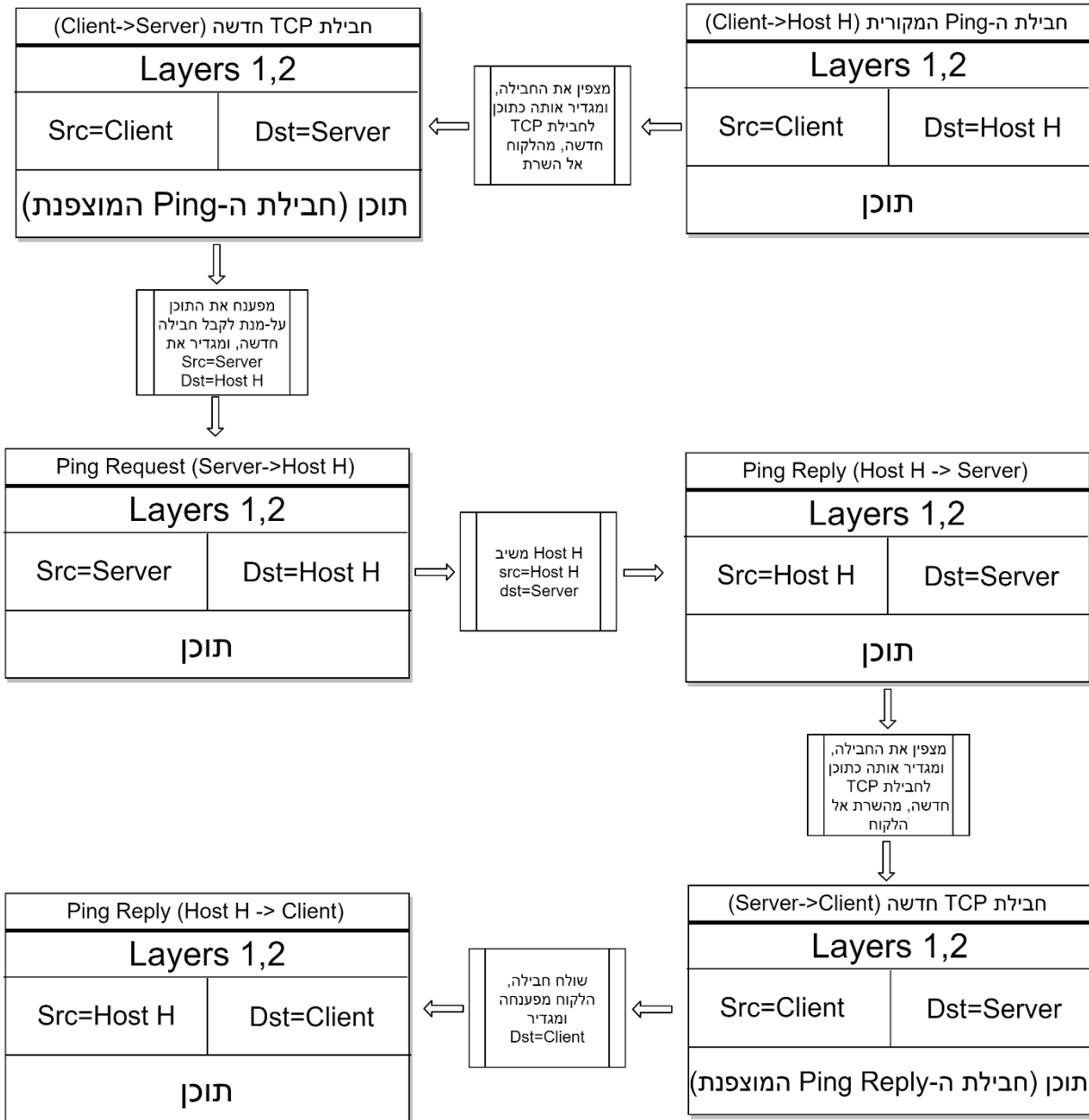
והיא מנתבת החוצה דרך התקן חיצוני Eth0 אל השרת, השרת מסיר את השכבות ומפענח את תוכן החבילה, שולח אותה החוצה והיא מנותבת אל היעד (כתובת המקור משתנה בדרך החוצה), החבילה מגיעה אל היעד והוא מחזיר אל המקור (השרת) תשובה-Ping Reply השרת מצפין את אותה ומגדיר אותה כתוכן לחבילה חדשה אותה הוא שולח ללקוח, הלקוח מסיר את השכבות ומפענח את תוכן החבילה כדי לקבל את החבילה שנשלחה מהיעד.



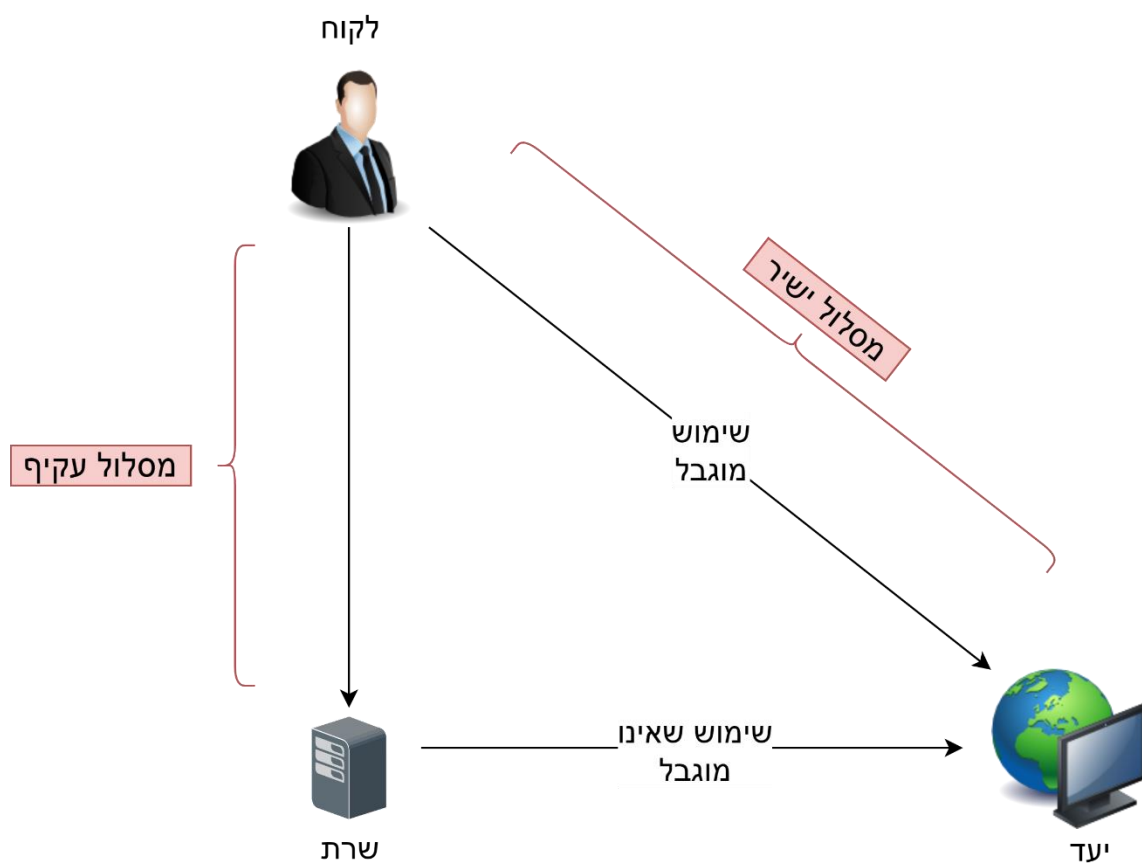
שינויים על החבילה מיציאה ועד חזרה:

## נתיב החבילה

מראה שינויים על החבילה לאורך  
נתיב מהלקוח אל השרת, מהשרת  
אל Host H  
ובחזרה מהשרת אל הלקוח



כמו בדוגמא שלעיל, אני מציג את השינויים על חבילות הרשת, ישנם שלושה גופים המוצגים בתרשים הם - **הלקוח (Server)**, **השרת (Client)** ו**היעד (Host H)** אליו נשלחת החבילה. הדוגמא מתחילה ב-Ping Request שמייצר הלקוח, אצלו היא מנותבת אל התקן ה-TUN, מוצפנת ע"י התוכנה ונשלחת בתוך חבילת TCP חדשה אל השרת, השרת מתיר את תוכן החבילה מפענח אותה ושולח אותה אל היעד

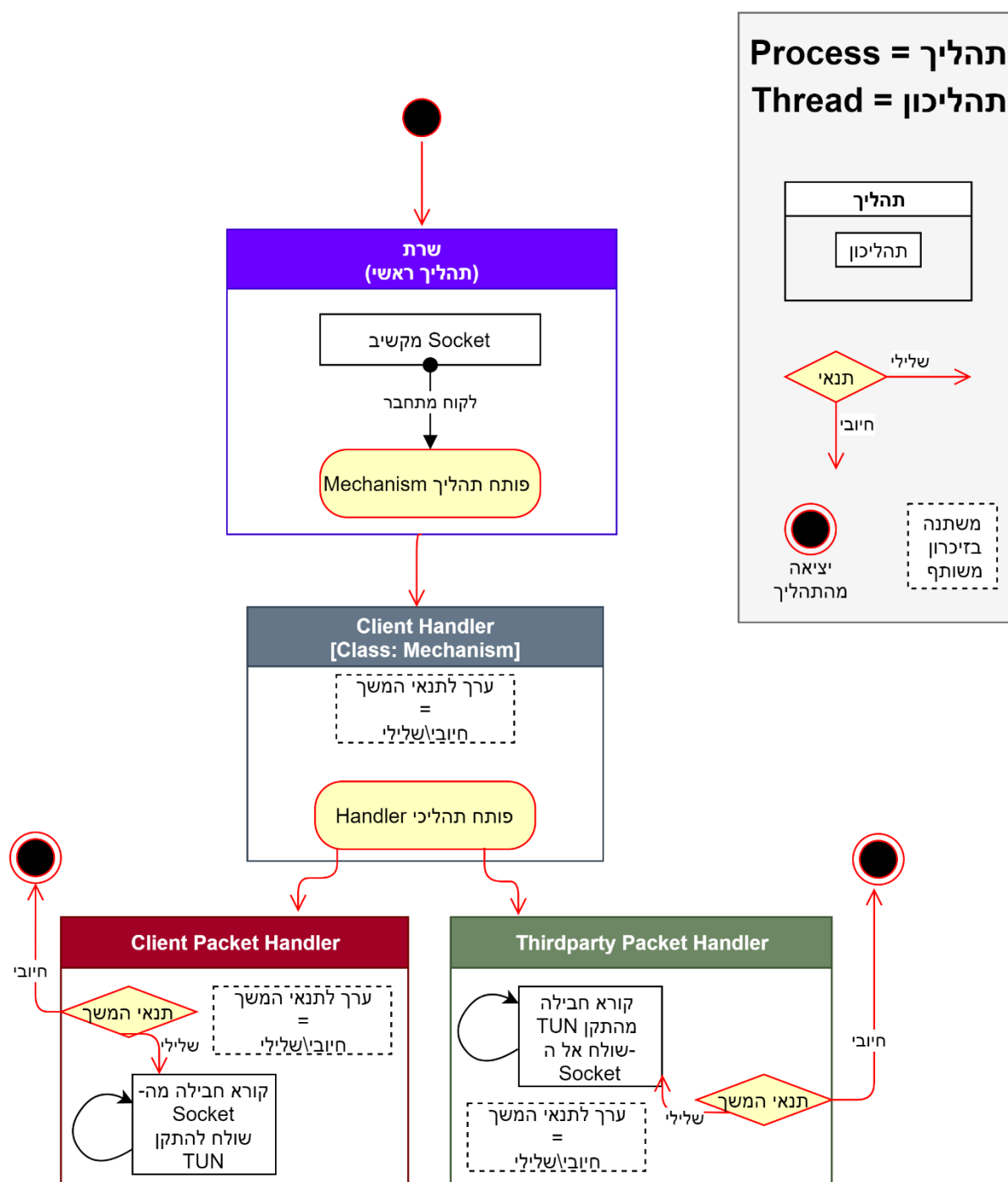


אם ולקוח מבקש להגיע ליעד מסוים אבל השירות שהוא מקבל הינו מוגבל בשל פרטי מיקומו/זהותו וכדומה, יכול הוא להעביר את הבקשות לשרת של המערכת וזה יעביר את הבקשות ליעד ויעביר את התשובות ללקוח.

## Activity Diagram:

לשני הצדדים אופי מבני דומה: בשניהם תהליך-אב שמוציא בהינתן אות תהליך Mechanism שאחראי להוציא תהליכים של קליטה מה-Socket וקליטה מהתקן ה-TUN, תהליכי הבן מושמדים כאשר ערך בזיכרון משותף (Multiprocessing.Value) – "תנאי עצירה" הופך לחיובי.

תרשים מבנה השרת:



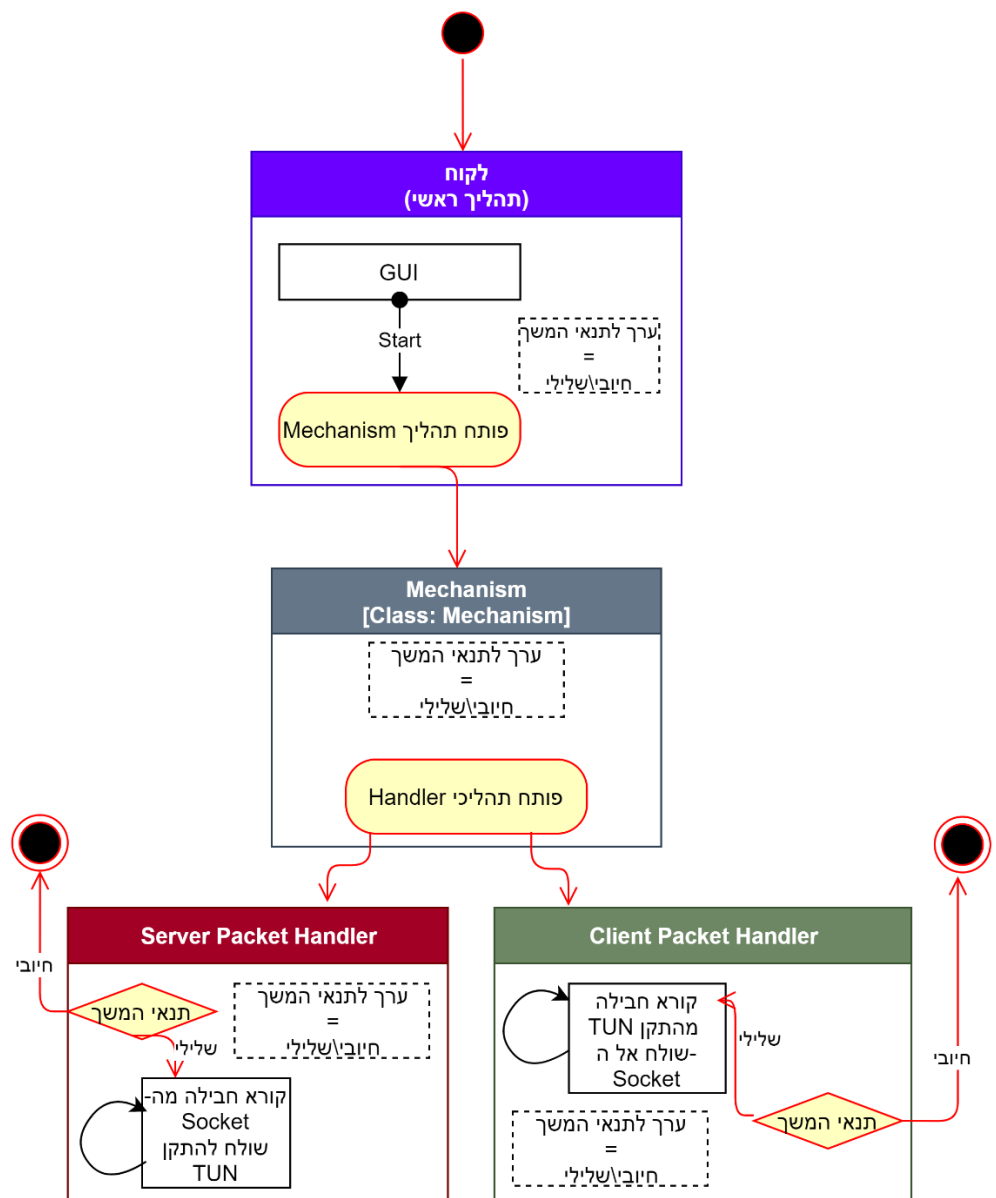
**התהליך הראשי** של השרת מקשיב לחיבורים מלקוחות, כשלקוח מתחבר השרת פותח תהליך חדש המטפל בלקוח, ומתפנה לקבלת לקוחות נוספים.

<sup>14</sup>**Multiprocessing.Value** – ערך בזיכרון משותף הניתן לקריאה מכל התהליכים, גישה בו-זמנית אליו אינה מובטחת, השתמשתי ב-Lock בכל קריאה של הערך

**תהליך Handler** מייצר שני תהליכים נוספים המטפלים בו זמנית בחבילות מהלקוח ומהיעד:

**האחד** - אחראי לקרוא מהלקוח את חבילה מה-Socket כלומר מהלקוח ולכתוב את החבילה את התקן ה-TUN, **השני** - אחראי לקרוא חבילה מהתקן ה-TUN ולכתוב אותה אל ה-Socket. מחלקת Mechanism מקצה (allocates) ערך מסוג **Multiprocessing.Value** המתנה לו אם יש להמשיך לרוץ, אם קיבל השרת הודעה מהלקוח שהוא מתנתק, או שגיאת Socket המעידה כי הלקוח התנתק, תנאי העצירה ישתנה, התהליכים יושמדו.

תרשים מבנה הלקוח:



**תהליך הראשי** של הלקוח אחראי על ממשק המשתמש, כשלקוח לוחץ "Start" התהליך פותח תהליך חדש המטפל בתקשורת עם השרת, שינויים בטבלת הניתוב, ויצירת התקן TUN.

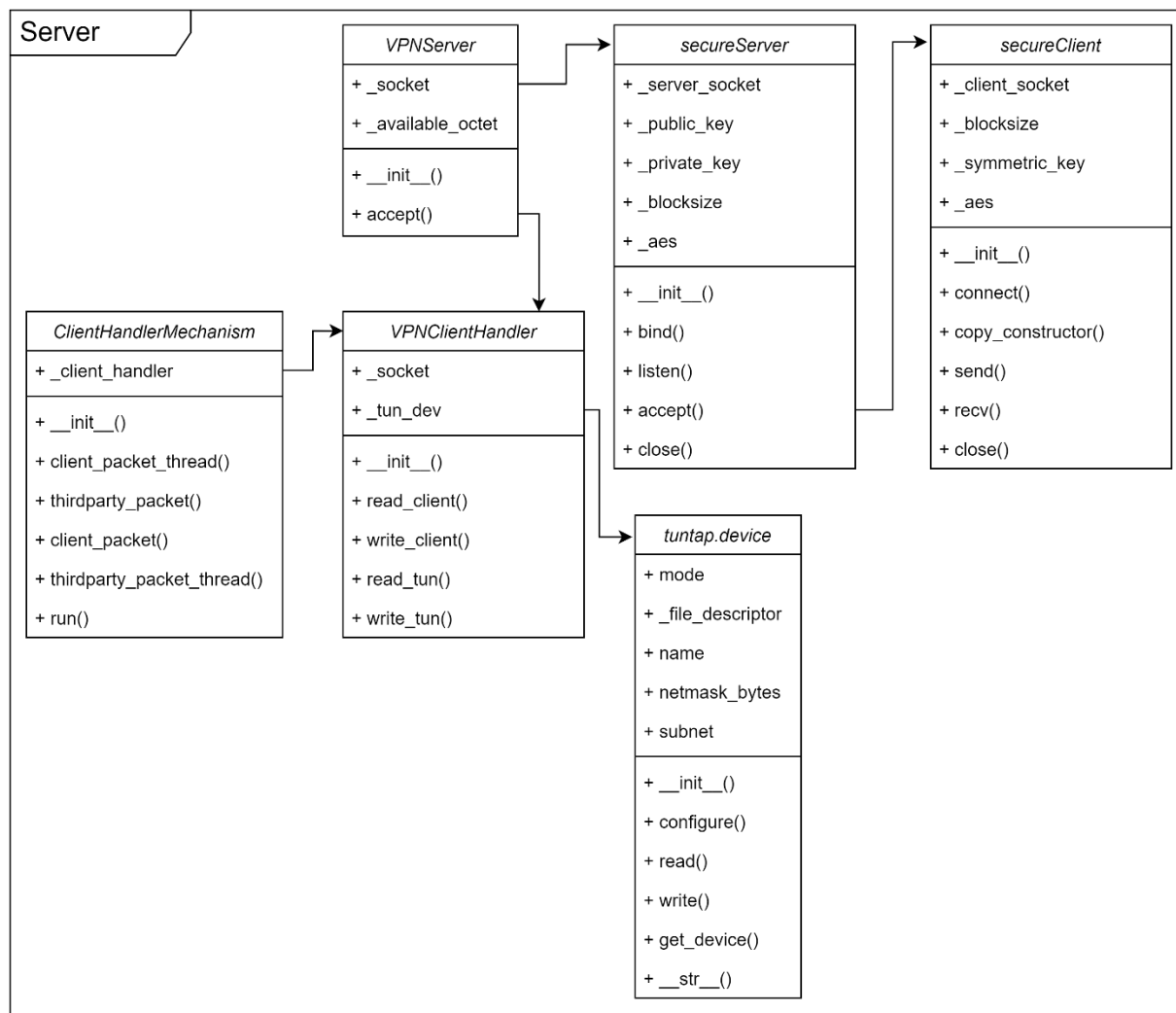
**Multiprocessing.Value** – ערך בזיכרון משותף הניתן לקריאה מכל התהליכים, גישה בו-זמנית אליו אינה מובטחת, השתמשתי ב-Lock בכל קריאה של הערך

**תהליך Handler** מייצר שני תהליכים נוספים המטפלים בו זמנית בחבילות מהלקוח עצמו ומהשרת:

**האחד** - אחראי לקרוא מהלקוח את חבילה מה-Socket כלומר מהשרת ולכתוב את החבילה את התקן TUN, **השני** - אחראי לקרוא חבילה מהתקן TUN ולכתוב אותה אל ה-Socket. מחלקת Mechanism מקצה (allocates) ערך מסוג **Multiprocessing.Value** המתנה לו אם יש להמשיך לרוץ, אם הלקוח לחץ "Stop", או שגיאת Socket המעידה כי הלקוח התנתק, תנאי העצירה ישתנה, התהליכים יושמדו.

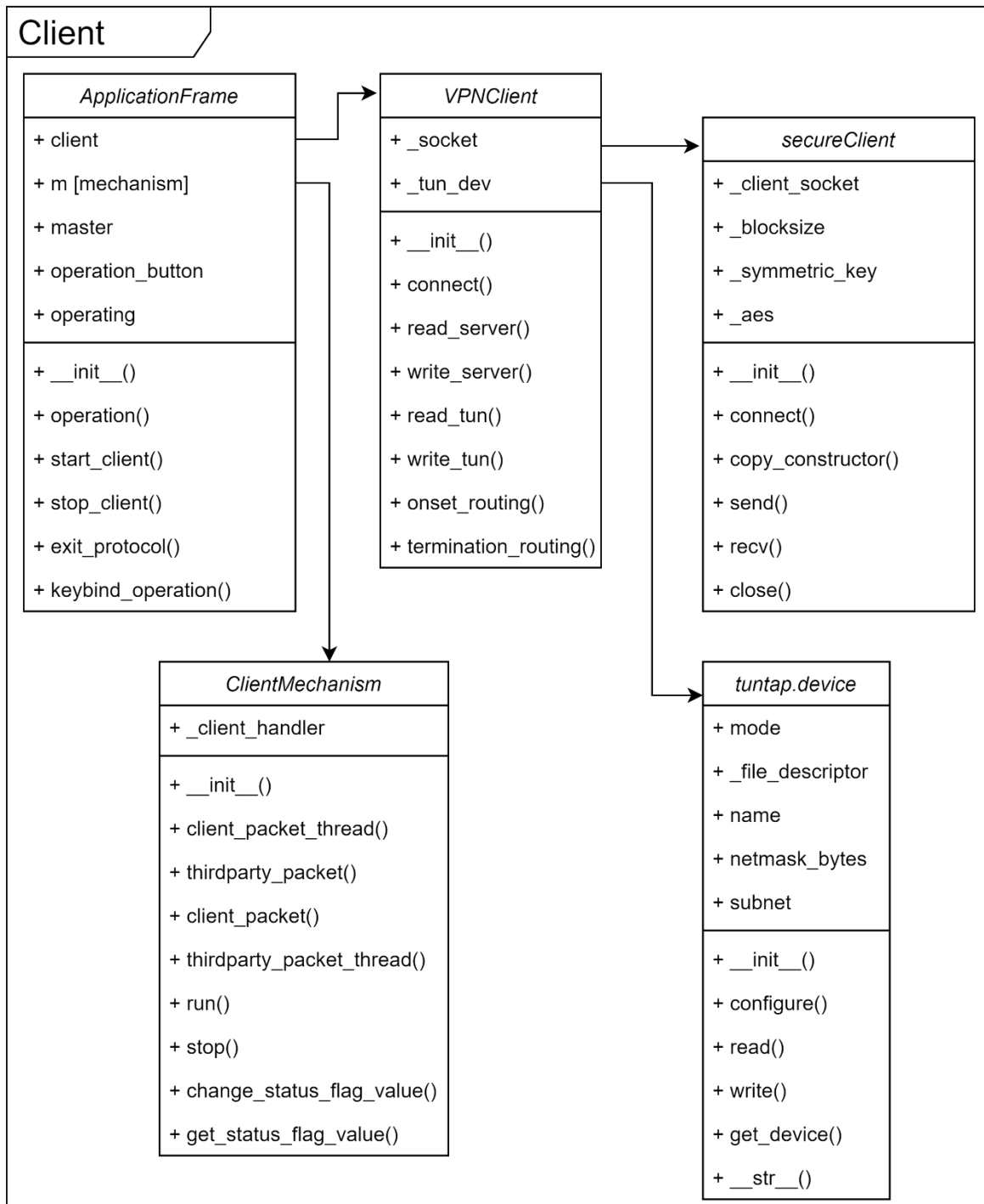
## Design Classes Diagram:

תרשים מבנה השרת:



VPNClientHandler – גוף אב למנוע השרת שמקבל secureClient ומוציא VPNClientHandler  
 ClientHandleMechanism – אחראי לחלוקת פעולות תקשורת עם הלקוח ועם היעד, בין שני תהליכים אותם הוא מייצר  
 VPNClientHandler – מאגד את כל הפעולות של התקשורת עם הלקוח והיעד  
 tuntap.device – מנגישה את השימוש בהתקן TUN  
 secureServer – מאגד את הרכיבים העברת המפתח של RSA, הצפנה עם AES, ותקשורת עם Socket לרכיב יחיד, עם פונקציות של שרת, הsocket הפסיבי.  
 secureClient – מאגד את הרכיבים העברת המפתח של RSA, הצפנה עם AES, ותקשורת עם Socket לרכיב יחיד, עם פונקציות של לקוח, הsocket האקטיבי.

תרשים מבנה הלקוח:



ApplicationFrame – מממשת פעולות של ספריית Tkinter, מפעילה את מנגנון הלקוח, ClientMechanism  
 ClientMechanism – אחראי לחלוקת פעולות תקשורת עם השרת ועם המערכת בין תהליכים  
 VPNClient – אחראי לתקשורת עם התקן TUN ועם השרת, מבצע שינויים בטבלת הניתוב ומבטל אותן  
 tuntap.device – מגישה את השימוש בהתקן TUN.

secureClient – מאגד את הרכיבים העברת המפתח של RSA, הצפנה עם AES, ותקשורת עם Socket לרכיב יחיד, עם פונקציות של לקוח, הsocket האקטיבי.

## טכנולוגיות בהן נעשה שימוש:

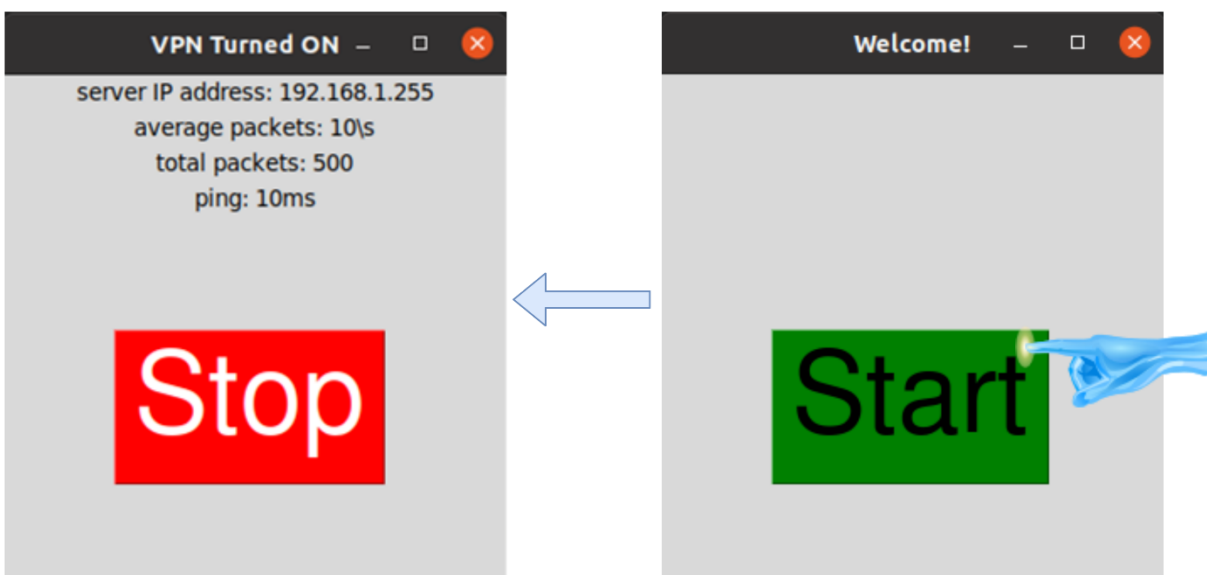
- המערכת רצה בתוך סביבת לינוקס (Ubuntu 20.04<sup>15</sup>)
- כל כולו של הקוד נכתב ב-Python גרסה 3.8<sup>16</sup>.
- ספריית Tkinter<sup>17</sup> בשביל גרפיקה (חלון, כפתורים, כותרות)
- ספריית rsa<sup>18</sup> בשביל ההצפנה הא-סימטרית
- ספריית os<sup>19</sup> בשביל קריאות אל המערכת, שימוש בהתקן TUN, שינוי בטבלת הניתוב
- ספריית PyCryptodome<sup>20</sup> בשביל ההצפנה הסימטרית בפרוטוקול AES
- ספריית Pickle<sup>21</sup> כדי להמיר את המפתח הציבורי למערך בתים ובחזרה

## API-ים:

- על מנת לבצע שינויים בהתקן TUN השתמשתי בספריית fcntl.ioctl המגשרת בין פייתון לספריית fcntl.ioctl בשפת C המשתמשת ב-Linux interface



## מדריך למשתמש:



השימוש מאוד פשוט, אפשר להתחיל ולכבות את המערכת בלחיצה על כפתור ה-Start, שלאחר מכן יהפוך ל-Stop.

אותה פעולה יכולה להתבצע גם עם קיצור-הדרך CTRL+S:



## מדריך למפתח:

צד השרת:

### VPNServer.py – VPNServer()

תכונות:

שם	סוג	הסבר	ערך התחלתי
<code>SERVER_INFO</code>	Tuple	מכיל את ה-IP והפורט שהשרת מקשיב עליהם	( <code>'0.0.0.0'</code> , <code>1111</code> )
<code>_socket</code>	<code>secureServer</code>	Socket עם פרטי הלקוח ומפתח הצפנה	<code>secureServer</code> חדש
<code>_available_octet</code>	String	מספר בין 1-255 של בית פנוי בכתובת IP	1

פעולות:

חתימה	פעילות	return
<code>__init__()</code>	יוצר <code>secureServer()</code> ומקשיב ללקוחות	None
<code>accept()</code>	מקבל <code>secureClient()</code> ואוקטטה פנויה	מחזיר <code>VPNClientHandler()</code>

```
'''
Author: Arie Gluzman
Creation Date: 22.3.2021

class VPNServer():
    encapsulation of use of rsa_server.secureServer() and
    VPNClientHandler.VPNClientHandler()
'''

import tuntap
import rsa_server
import rsa_client
import VPNClientHandler

# CONSTANTS #
DEFAULT_HOST_IP = '0.0.0.0'
DEFAULT_PORT = 1111
SERVER_INFO = (DEFAULT_HOST_IP, DEFAULT_PORT)
MAX_CLIENTS_WAITING = 1
# END #

class VPNServer:

    def __init__(self):
        self._socket = rsa_server.secureServer()
        self._socket.bind(SERVER_INFO)
        self._socket.listen(MAX_CLIENTS_WAITING)
        self._available_octet = 1 # the ending octet of a subnet given to
the TUN device value between 1-255
                                # e.g, '1.1.1.1' then '1.1.1.2' .....
                                '1.1.1.255'
```

```
def accept(self): # returns a client handler with the secure socket and
    available address for TUN device
    self._available_octet += 1 # use of an address
    print('10.0.8.'+str(self._available_octet))
    return VPNClientHandler.VPNClientHandler(self._socket.accept(),
    str(self._available_octet - 1))
```

## VPNClientHandler.py – VPNClientHandler()

### תכונות

שם	סוג	הסבר	ערך התחלתי
<code>_socket</code>	<code>secureClient</code>	Socket עם פרטי הלקוח ומפתח הצפנה	<code>init()</code> מתקבל ב
<code>_tun_dev</code>	<code>tuntap.device()</code>	התקן TUN	<code>tuntap.device()</code> חדש

### פעולות

חתימה	פעילות	return
<code>__init__(secureClient(), string)</code>	מייצר ומגדיר <code>tuntap.device()</code>	None
<code>read_client()</code>	קורא מה-Socket מידע	Bytes
<code>write_client(Bytes)</code>	כותב אל ה-Socket	None
<code>read_tun()</code>	קורא מה- <code>tuntap.device()</code>	Bytes
<code>write_tun(Bytes)</code>	כותב אל ה- <code>tuntap.device()</code>	None

```
'''
Author: Arie Gluzman
Creation Date: 22.3.2021

class VPNClientHandler():
    a collection of attributes and functions for communication with the
    client and the external host
'''

import rsa_client
import tuntap

class VPNClientHandler:

    def __init__(self, securesocket, available_octet):
        if type(available_octet) is not str:
            raise ValueError('')
        elif not available_octet.isnumeric():
            raise ValueError('')
        elif not 0 <= int(available_octet) <= 255:
            raise ValueError('Last octet %s is invalid, must be between 0-
255' % available_octet)
        self._socket = securesocket
        self._tun_dev = tuntap.device(tuntap.IFF_NO_PI | tuntap.IFF_TUN,
b'tun_dev%d') # TUN device with the
```

```

next available name of type 'tun_dev%d'
# e.g, 'tun_dev0'
self._tun_dev.configure('10.0.8.' + available_octet, 24) #
configuration of device on the system
print('DEVICE NAME: ' + self._tun_dev.name)

def read_client(self):
    return self._socket.recv()

def write_client(self, data):
    self._socket.send(data)

def read_tun(self):
    return self._tun_dev.read(1500)

def write_tun(self, data):
    self._tun_dev.write(data)

```

### tuntap.py – device()

#### תכונות

שם	סוג	הסבר	ערך התחלתי
<i>TUNSETIFF, IFF_TUN, IFF_TAP, IFF_NO_PI</i>	שרשרת בתים הקסדצימלית	דגלי הגדרות להתקן TAP\TUN	0x400454ca, 0x0001, 0x0002, 0x1000
<i>_file_descriptor</i>	File descriptor	שם הקובץ לפניה למכשיר הTUN	os.open('dev/net/tun', os.O_RDWR)
name	String	שם המכשיר הרשמי	"tun_dev0", "1", "2", "3" .....
<i>netmask_bytes</i>	int	מספר הבתים ב-subnet	-1
<i>subnet</i>	String	כתובת המכשיר	מתקבלת ב-__init__()

#### פעולות

חתימה	פעילות	return
<i>__init__(hexBytes, Bytes)</i>	קריאות מערכת ליצירת התקן TUN	None
<i>configure(String, Int)</i>	מגדיר את ערכי subnet ומבצע קישור של התקן הTUN בהגדרות המערכת	Bytes
<i>read(Bytes)</i>	קורא מההתקן	Bytes
<i>write(Bytes)</i>	כותב אל ההתקן	Bytes
<i>get_device()</i>	מחזיר את <i>_file_descriptor</i>	File Object
<i>__str__()</i>	מחזיר שרשרת עם פרטיו של ההתקן, שם, דגלים וכו'.	String

```

'''
Author: Arie Gluzman
Creation Date: 13.1.2021
'''

```

```

for further explanation on tun device -
https://www.freebsd.org/cgi/man.cgi?query=tun&sektion=4

'''

import os
import fcntl
import struct

# TUN/TAP device flags #
TUNSETIFF = 0x400454ca
IFF_TUN = 0x0001
IFF_TAP = 0x0002
IFF_NO_PI = 0x1000
'''

Flags:
TUNSETIFF - Multi-Queue can be called using same file descriptor
IFF_TUN - layer 3, TUN device (no Ethernet headers)
IFF_TAP - layer 2, TAP device
IFF_NO_PI - Do not provide packet information
'''

# end #

class device:

    def __init__(self, mode, name=b'tun%d'):
        '''
        :param mode: hex array containing the composition of flags
        :param name: byte array containing format string of the name
        '''
        self.mode = mode
        if type(name) != bytes:
            raise ValueError("'name' must be a byte array type, not %s." %
type(name))

        # Create the tun interface
        self._file_descriptor = os.open("/dev/net/tun", os.O_RDWR)
        ifr = struct.pack('16sH', name, mode) # Create C struct of flags
(device mode) and device name
        ifname_bytes = fcntl.ioctl(self._file_descriptor, TUNSETIFF,
ifr) # send struct to Kernel to
manipulate device-at file-descriptor, name and mode
        ifname = ifname_bytes.decode('UTF-8')[:16].strip("\x00") # Get
Device Name

        self.name = ifname
        self.configured = False
        self.netmask_len = -1 # Initialize Value
        self.subnet = '' # Initialize Value

    def configure(self, subnet, netmask_len): # system configuration of
device
        if type(netmask_len) != int:
            raise ValueError("'netmask_len' must be of type Integer, not %s."

```

```

% type(netmask_len))
    if not (33 > netmask_len > 0):
        raise ValueError(
            "'netmask_len' value - '%d' is out of bounds, must be between
0 to 32." % netmask_len)

    self.subnet = subnet
    self.netmask_len = netmask_len

    # set up the tun interface
    os.system(
        "sudo ip addr add {}/{}/{} dev {}".format(self.subnet,
self.netmask_len, self.name)) # add address to device
    os.system("sudo ip link set dev {} up".format(self.name)) # set
device 'up'
    self.configured = True

    def deconfigure(self): # undoes system configuration at self.configure()
        if not self.configured:
            raise Exception('configure() must be used prior to
deconfigure()')
        os.system("sudo ip addr del {}/{}/{} dev {}".format(self.subnet,
self.netmask_len, self.name))
        os.system("sudo ip link set dev {} down".format(self.name))

    def terminate(self): # used at end of use
        os.close(self._file_descriptor) # disables use of file descriptor,
releasing related resources

    def read(self, bts):
        if type(bts) != int:
            raise ValueError("'bts' must be of type Integer, not %s." %
type(bts))
        return os.read(self._file_descriptor, bts)

    def write(self, msg):
        if type(msg) != bytes:
            raise ValueError("'msg' must be of type Bytes, not %s." %
type(msg))
        return os.write(self._file_descriptor, msg)

    def get_device(self): # returning the file-descriptor
        return self._file_descriptor

    def __str__(self):
        return 'Flags: {}, Name: {}\\nconfiguration:
{}/{}/{}'.format(hex(self.mode), self.name, self.subnet,
self.netmask_len)

```

**rsa\_server.py – secureServer()**

תכונות

שם	סוג	הסבר	ערך התחלתי
<code>_server_socket</code>	<code>Socket.socket</code>	<code>Socket</code> שרת בפרוטוקול <code>TCP</code>	<code>Socket</code> חדש

rsa.newkeys(512)	זוג אובייקטים, מפתח הצפנה פרטי וציבורי. באורך 512 בתים	PublicKey, PrivateKey	<code>_public_key</code> , <code>_private_key</code>
ברירת מחדל 32	אורך בלוק להצפנה	Int	<code>_blocksize</code>
None	אובייקט המצפין באמצעות פרוטוקול AES	Crypto.Cipher.AES	<code>_aes</code>

#### פעולות

return	פעילות	חתימה
None	מגדיר את ה-Socket, את מפתחות ההצפנה ואת אובייקט ההצפנה	<code>__init__(AddressFamily, SocketKind)</code>
None	קשירת ה-Socket אל כתובתו	<code>bind(Tuple)</code>
None	מגדיר ה-Socket כפסיבי, המחכה לחיבור, מגדיר את אורך התור בבקשה לחיבור	<code>listen(Int)</code>
Socket.socket	מייצא את החיבור אל הלקוח הראשון בתור	<code>accept()</code>
None	סוגר את ה-File descriptor ואת המשאבים הקשורים אליו	<code>close()</code>

```
'''
Author: Arie Gluzman
Creation Date: 1.12.2020

class secureServer():
    an encapsulation of RSA key transfer, AES encryption and socket
    connection and function.
    by using connect(), send() and receive() the user is implementing all
    three components.
'''

from rsa_client import secureClient
from Crypto.Cipher import AES
from Crypto.Util.Padding import pad, unpad
from pickle import dumps as pickle_dumps
import rsa
import socket

class secureServer:

    def __init__(self, net_protocol=socket.AF_INET,
transport_protocol=socket.SOCK_STREAM,
blocksize=32): # defaults are set to an IPv4, TCP socket.
Encryption Block-Size 32
        self.server_socket = socket.socket(net_protocol, transport_protocol)
        self._public_key, self._private_key = rsa.newkeys(512) # generating
RSA public, private keys
        self._blocksize = blocksize
        self._aes = None

    def bind(self, arguments):
        if type(arguments) not in (tuple, list):
```

```

        raise TypeError('argument given is not a tuple/string.')
    ip, port = arguments[0], arguments[1]
    '''
    :param ip:
    :param port:
    '''
    if type(ip) is not str:
        raise ValueError("IP value given is invalid %s is not 'str'." %
ip)
    if type(port) is not int:
        raise ValueError("PORT value given is invalid %s is not 'int'." %
port)
    self._server_socket.bind((ip, port))

    def listen(self, clients): # setting maximum client connection request
queue length
        if type(clients) is not int:
            raise TypeError("argument must be of type 'str' not %s." %
type(clients))
        self._server_socket.listen(clients)

    def accept(self): # establishing connection, getting AES symmetric key
and returning secureClient() for further use
        try:
            client_socket, addr = self._server_socket.accept()
        except socket.error as e:
            raise socket.error('An Error Occurred while trying to receive
connection .' + e)
        pickled_pubkey = pickle_dumps(self._public_key) # turning object
into byte-array
        client_socket.send(pickled_pubkey) # sending public key to client
        encrypted_symkey = client_socket.recv(4000) # receiving an encrypted
byte array of the symmetric key
        symmetric_key = rsa.decrypt(encrypted_symkey, self._private_key) #
decrypting byte array for symmetric key
        self._aes = AES.new(symmetric_key, AES.MODE_ECB) # Crypto.Cipher.AES
for encryption with symmetric key
        clt = secureClient()
        clt.copy_constructor(client_socket, symmetric_key) #
rsa_client.secureClient with symmetric key and socket
        return clt

    def close(self): # run at end of use
        self._server_socket.close() # releasing file-descriptor related
resources

```

### rsa\_client.py – secureClient()

#### תכונות

שם	סוג	הסבר	ערך התחלתי
<code>_client_socket</code>	<code>Socket.socket</code>	<code>Socket</code> בפרוטוקול TCP	<code>Socket.socket</code> חדש
<code>_blocksize</code>	<code>Int</code>	גודל הבלוק/מפתח ההצפנה	32



מערך בתים אקראי	מספר בתים אקראי בגודל _blocksize	Bytes	_symmetric_key
Crypto.Cipher.AES חדש	אובייקט מצפין AES	Crypto.Cipher.AES	_aes

#### פעולות

return	פעילות	חתימה
None	מיצר socket.socket	__init__()
Bytes	קורא מה-Socket מידע	connect()
None	מייצר rsa_client.secureClient עם תכונות נתונות	copy_constructor(socket.socket, Bytes)
None	שולח בתים, מוצפנים עם _aes דרך ה-Socket	send(Bytes)
Bytes	קורא בתים דרך ה-Socket ומפענחם עם _aes	recv()
None	סוגר את ה-File Descriptor ומשחרר את המשאבים הכפופים אליו	close()

```
'''
Author: Arie Gluzman
Creation Date: 1.12.2020

class secureClient():
    an encapsulation of RSA key transfer, AES encryption and socket
    connection and function.
    by using connect(), send() and receive() the user is implementing all
    three components.
'''

from Crypto.Random import get_random_bytes as generate_key
from Crypto.Cipher import AES
from Crypto.Util.Padding import pad, unpad
from pickle import loads as pickle_loads
import rsa
import socket

default_blocksize = 32

class secureClient:

    def __init__(self, net_protocol=socket.AF_INET,
transport_protocol=socket.SOCK_STREAM):
        # defaults are set to an IPv4, TCP socket. Encryption Block-Size 32
        self._client_socket = socket.socket(net_protocol, transport_protocol)

    def connect(self, arguments):
        if type(arguments) not in (tuple, list):
            raise Exception('argument given is not a tuple or list')
        if not len(arguments) == 2:
            raise Exception('Arguments Error. secureClient requires 2-3
arguments, %s were given.' % len(arguments))
```

```

ip, port = arguments[0], arguments[1]
'''
:param ip:
:param port:
:return:
'''
if type(ip) is not str:
    raise ValueError("IP value given is invalid %s is not 'str'." %
ip)
if type(port) is not int:
    raise ValueError("PORT value given is invalid %s is not 'int'." %
port)

self._ip = ip
self._port = port
self._blocksize = default_blocksize
self._symmetric_key = generate_key(self._blocksize) # generates
blocksize i.e 32 random bytes for symmetric key
self._aes = AES.new(self._symmetric_key, AES.MODE_ECB) #
Crypto.Cipher.AES for encryption with symmetric key
try:
    self._client_socket.connect((self._ip, self._port))
except socket.error as e:
    raise socket.error('An error occurred while trying to connect to
server: ' + e)
pickled_pubkey = self._client_socket.recv(4000) # receives public
key object as byte-array
public_key = pickle.loads(pickled_pubkey) # loads public key byte-
array object
self._client_socket.send(rsa.encrypt(self._symmetric_key,
public_key)) # sends encrypted symmetric key

def copy_constructor(self, client_socket, symkey):
    '''
    :param client_socket:
    :param symkey:
    used when socket and symmetric key are given before-hand and not by
using connect()
    used at server after which receives symmetric key from client
    :return: None
    '''
    self._client_socket = client_socket
    self._symmetric_key = symkey
    self._blocksize = default_blocksize
    self._started = True
    self._aes = AES.new(self._symmetric_key, AES.MODE_ECB)

def send(self, bytes):
    '''
    :param bytes: byte-array to be send
    :return: None

    padding data, adding its character length, and character-length
length, encrypting and sending.
    for message b'hello', text=b'hello', len='5', len-of-len='1':
b'15hello'
    that way client knows how much to receive, and can differentiate

```

```

between messages.
in this case message can load up to about 10mb.
'''
message = pad(bytes, self._blocksize) # max size 10mb
length = str(len(message)) # max 9 characters
len_of_length = str(len(length)) # max 1 character
self._client_socket.send((len_of_length + length).encode() +
self._aes.encrypt(message))

def recv(self):
'''
:return: plain-text byte array
gets length of length-of-data, length-of-data, and data.
'''
len_of_length = int(self._client_socket.recv(1).decode())
length = int(self._client_socket.recv(len_of_length).decode())
return unpad(self._aes.decrypt(self._client_socket.recv(length)),
self._blocksize)

def close(self): # used at the end of use
self._client_socket.close()

```

## routing.py

### תכונות

שם	סוג	הסבר	ערך התחלתי
INSERT	String	יכנס לפקודה אם ערך הboolean ב-set_route() חיובי	'add'
REMOVE	String	יכנס לפקודה אם ערך הboolean ב-set_route() שלילי	'del'

### פעולות

חתימה	פעילות	Return
set_route(Boolean, String, Int, String, String)	מכניס לטבלת הניתוב ערך ע"פ הנתונים	None

```

'''
Author: Arie Gluzman
Creation Date: 20.3.2021
'''

import os

INSERT = 'add' # insert route entry
REMOVE = 'del' # remove route entry

def set_route(operation, subnet, netmask, gateway, device):
'''

```

```

:param operation: Boolean, determines whether function is to add or to
remove an entry
:param subnet: String, destination subnet or address
:param netmask: Integer, 0-32, destination subnet bytes used (32 if full
address is given)
:param gateway: String, address to which packet is to be routed
:param device: String, device name to which packet is to be routed
:return: None

In essence, with parameters, the function constructs a string which it
writes to OS Kernel
root@user$ sudo route *operation* *subnet*/*netmask* gw*gateway* dev
*device*
e.g, root@user$ sudo route add 0.0.0.0/0 gw 192.168.1.1 dev enp0s3
'''
if type(operation) is not bool:
    raise ValueError("value must be of type 'bool', an argument of type
{} was given".format(type(operation)))
operation_text = REMOVE
if operation:
    operation_text = INSERT
print('sudo route {} {}/{ gw {} dev {}'.format(operation_text, subnet,
netmask, gateway, device))
os.system('sudo route {} {}/{ gw {} dev {}'.format(operation_text,
subnet, netmask, gateway, device))

```

### server\_mechanism.py – ClientHandlerMechanism(multiprocessing.Process)

#### תכונות

שם	סוג	הסבר	ערך התחלתי
_client_handler	rsa_client.secureClient	ה-Socket המאובטח שהתחבר אל השרת	מתקבל ב(__init__)

#### פעולות

חתימה	פעילות	return
__init__()	מפעילה את איתחול ה-multiprocessing.Process()	None
client_packet_thread()	לולאה שמפעילה Thread של client_packet(), עד אות סיום	None
thirdpacket_packet_thread()	לולאה שמפעילה Thread של thirdparty_packet(), עד אות סיום	None
client_packet()	קורא מהלקוח, כותב אל התקן TUN	None
thirdparty_packet()	קורא מהתקן TUN, כותב אל הלקוח	None
run()	מוציא תהליכים של client_packet_thread() ו-client_packet_thread()	None

```

'''
Author: Arie Gluzman
Creation Date: 25.3.2021

class CliendHandleMechanism():
    responsible for communication with destination 'host H' and client,
    and for distribution of work between processes and threads for it's
    purpose.
'''

```

```
import threading
import multiprocessing

STOP_MESSAGE = 'STOP'.encode()

class ClientHandleMechanism(multiprocessing.Process):

    def __init__(self, client_handler):
        multiprocessing.Process.__init__(self)
        self._client_handler = client_handler

    def client_packet_thread(self): # creates loop, thread for
client_packet()
        while True:
            c = threading.Thread(target=self.client_packet)
            c.start()
            c.join()

    def thirdparty_packet(self): # creates loop, thread for
thirdparty_packet_thread()
        while True:
            t = threading.Thread(target=self.thirdparty_packet_thread)
            t.start()
            t.join()

    def client_packet(self): # reads packet from client, writes to TUN
device
        client_pack = self._client_handler.read_client()
        if client_pack == STOP_MESSAGE:
            print('CLIENT HINTED STOP')
        else:
            self._client_handler.write_tun(client_pack)

    def thirdparty_packet_thread(self): # reads packet from TUN device,
writes to client
        thdp_packet = self._client_handler.read_tun()
        self._client_handler.write_client(thdp_packet)

    def run(self) -> None:
        c = multiprocessing.Process(target=self.client_packet_thread)
        t = multiprocessing.Process(target=self.thirdparty_packet_thread)
        c.start()
        t.start()
```

צד הלקוח (משתמש במחלקות `secureClient()`, `tuntap.device()`, ובפונקציה `set_route()`):

#### client\_mechanism.py – ClientMechanism(multiprocessing.Process):

תכונות

שם	סוג	הסבר	ערך התחלתי
<code>STANDARD</code> , <code>USER_STOP</code> , <code>SOCKET_ERROR</code>	Int	ערכי <code>_status_flag</code> המביעים מצב	0,1,2 בהתאמה

socket.socket חדש	Socket בפרוטוקול TCP	socket.socket	<code>_client</code>
0	ערך מספרי המספר על מצב המערכת	<code>multiprocessing.Value</code>	<code>_status_flag</code>
Multiprocessing.Lock חדש	אובייקט <code>multiprocessing.Lock</code> המגדיר גישה ל- <code>_status_flag</code>	<code>multiprocessing.Lock</code>	<code>_status_flag_lock</code>
Multiprocessing.Process חדש	אובייקט <code>multiprocessing.Process</code> המפעיל את- <code>client_packet_thread()</code>	<code>multiprocessing.Process</code>	<code>client_process</code>
Multiprocessing.Process חדש	אובייקט <code>multiprocessing.Process</code> המפעיל את- <code>thirdpacket_packet_thread()</code>	<code>multiprocessing.Process</code>	<code>thdp_process</code>

#### פעולות

return	פעילות	חתימה
None	מיצר <code>socket.socket</code>	<code>__init__()</code>
None	מאתחל את המשתנים ואת פעולת ה- <code>__init__()</code> של <code>multiprocessing.Process</code>	<code>connect()</code>
None	קורא מה-Socket וכותב אל התקן ה-TUN	<code>thirdparty_packet()</code>
None	קורא מהתקן ה-TUN וכותב אל ה-Socket	<code>client_packet()</code>
Bytes	לולאה המייצרת <code>Thread</code> ים של <code>thirdparty_packet()</code> כל עוד <code>_status_flag</code> מרשה זאת	<code>thirdparty_packet_thread()</code>
None	לולאה המייצרת <code>Thread</code> ים של <code>client_packet()</code> כל עוד <code>_status_flag</code> מרשה זאת	<code>client_packet_thread()</code>
None	מייצר <code>Process</code> ים של <code>thirdparty_packet_thread()</code> ו- <code>client_packet_thread()</code>	<code>run()</code>
None	משנה את ערך <code>_status_flag</code> ל- <code>USER_STOP</code>	<code>stop()</code>
None	משנה ערכו של <code>_status_flag</code> (עוצר את הגישה ל- <code>_status_flag_lock</code> באמצעות)	<code>change_status_flag_value()</code>
Int	מחזיר ערכו של <code>_status_flag</code> (עוצר את הגישה ל- <code>_status_flag_lock</code> באמצעות)	<code>get_status_flag_value()</code>

```
'''
Author: Arie Gluzman
Creation Date: 25.3.2021

class CliendHandleMechanism():
    responsible for communication with destination 'host H' and client,
    and for distribution of work between processes and threads for it's
    purpose.
'''

import threading
import multiprocessing

STOP_MESSAGE = 'STOP'.encode()
```

```
class ClientHandleMechanism(multiprocessing.Process):

    def __init__(self, client_handler):
        multiprocessing.Process.__init__(self)
        self._client_handler = client_handler

    def client_packet_thread(self): # creates loop, thread for
client_packet()
        while True:
            c = threading.Thread(target=self.client_packet)
            c.start()
            c.join()

    def thirdparty_packet(self): # creates loop, thread for
thirdparty_packet_thread()
        while True:
            t = threading.Thread(target=self.thirdparty_packet_thread)
            t.start()
            t.join()

    def client_packet(self): # reads packet from client, writes to TUN
device
        client_pack = self._client_handler.read_client()
        if client_pack == STOP_MESSAGE:
            print('CLIENT HINTED STOP')
        else:
            self._client_handler.write_tun(client_pack)

    def thirdparty_packet_thread(self): # reads packet from TUN device,
writes to client
        thdp_packet = self._client_handler.read_tun()
        self._client_handler.write_client(thdp_packet)

    def run(self) -> None:
        c = multiprocessing.Process(target=self.client_packet_thread)
        t = multiprocessing.Process(target=self.thirdparty_packet_thread)
        c.start()
        t.start()
```

#### application.py – ApplicationFrame(tkinter.Frame):

תכונות

שם	סוג	הסבר	ערך התחלתי
<i>client</i>	VPNClient.VPNClient	מנהל התקשורת וההצפנה	None
<i>master</i>	tkinter.TK	אחראי לגרפיקה	tkinter.Tk()
<i>operation_button</i>	tkinter.Button	כפתור התחלה, STOP\START	tkinter.Button()
<i>operating</i>	Boolean	ערך המורה אם המערכת פועלת או לא	False

פעולות

חתימה	פעילות	return
-------	--------	--------

None	מיצר את כל תכונות הגרפיקה, משתנים, עצמים שונים, קיצורי-דרך וכו'.	<code>__init__()</code>
None	מפעיל\מכבה את פעילות המערכת.	<code>operation()</code>
None	מתחיל את פעילות המערכת, מבצע שינויים בטבלת הניתוב.	<code>start_client()</code>
None	מכבה את פעילות המערכת, מחזיר שינויים בטבלת הניתוב לקדמותם.	<code>stop_client()</code>
None	מייצר חלון של וידוא היציאה, מפעיל את <code>stop_client()</code>	<code>exit_protocol()</code>
None	מפעיל את <code>operation()</code> אם הופעל הקיצור-דרך Ctrl+S	<code>keybind_operation()</code>

```
'''
Author: Arie Gluzman
Creation Date: Unknown, 2021

class ApplicationFrame():
    responsible for the Graphics User-Interface (GUI), and for the Initiation
    and Termination of Client Mechanism
'''

import tkinter.messagebox
from tkinter import Frame, Button, Tk, font, Label, messagebox
import VPNClient
import client_mechanism
import sys

# Constants #
DEFAULT_TITLE = 'Welcome!'
MODE_ON_TITLE = 'VPN Turned ON'
MODE_OFF_TITLE = 'VPN Turned OFF'

DEFAULT_BUTTON_TEXT_PADDING_X = 10
DEFAULT_BUTTON_TEXT_PADDING_Y = 10

MODE_ON_BUTTON_TEXT = 'Stop'
MODE_OFF_BUTTON_TEXT = 'Start'

DEFAULT_BUTTON_FONT = None

MODE_ON_BUTTON_BACKGROUND_COLOR = 'red'
MODE_OFF_BUTTON_BACKGROUND_COLOR = 'green'
MODE_ON_BUTTON_FOREGROUND_COLOR = 'white'
MODE_OFF_BUTTON_FOREGROUND_COLOR = 'black'

DEFAULT_SIZE = '300x300'

DEFAULT_SERVER_IP = '192.168.1.223'
DEFAULT_PORT = 1111
SERVER_INFO = (DEFAULT_SERVER_IP, DEFAULT_PORT)

# Constants #

class ApplicationFrame(Frame):
    def __init__(self, master): # creates all graphical assets, constants,
        variables and other assets
```



```

        self.client = None
        self.m = None
        self.server_ip = '192.168.1.255'
        self.ping = '10'
        self.avg_packets = '10'
        self.total_packets = '500'
        self.master = master
        self.master.protocol('WM_DELETE_WINDOW', self.exit_protocol)
        self.master.geometry(DEFAULT_SIZE)
        self.master.title(DEFAULT_TITLE)
        self.operation_button = Button(master, text=MODE_OFF_BUTTON_TEXT,
command=self.operation,
                                bg=MODE_OFF_BUTTON_BACKGROUND_COLOR,
fg=MODE_OFF_BUTTON_FOREGROUND_COLOR,
                                padx=DEFAULT_BUTTON_TEXT_PADDING_X,
pady=DEFAULT_BUTTON_TEXT_PADDING_Y)
        self.operation_button.place(x=65, y=150) # setting default location
for OPERATION button
        self.master.bind('<Control-s>',
                                self.keybind_operation) # creates a bind for
Ctrl+S, as alternative to Operation button
        DEFAULT_BUTTON_FONT = font.Font(family='Helvetica', size=50)
        self.operation_button['font'] = DEFAULT_BUTTON_FONT
        self.server_ip_label = Label(self.master, text='server IP address: '
+ self.server_ip)
        self.ping_label = Label(self.master, text='ping: ' + self.ping +
'ms')
        self.avg_packets_label = Label(self.master, text='average packets: '
+ self.avg_packets + '\\s')
        self.total_packets_label = Label(self.master, text='total packets: '
+ self.total_packets)
        self.server_ip_label.pack()
        self.avg_packets_label.pack()
        self.total_packets_label.pack()
        self.ping_label.pack()
        self.operating = False

    def operation(self): # function which runs when OPERATION button is
pressed, switches self.operating every press
        if self.operating: # when operation button is pressed and program is
running, meaning - STOP
            self.operation_button.configure(text=MODE_OFF_BUTTON_TEXT,
bg=MODE_OFF_BUTTON_BACKGROUND_COLOR,
fg=MODE_OFF_BUTTON_FOREGROUND_COLOR)
            self.master.title(MODE_OFF_TITLE)
            self.operating = False
            self.stop_client()
        else: # when operation button is pressed and program is not running,
meaning - START
            self.operation_button.configure(text=MODE_ON_BUTTON_TEXT,
bg=MODE_ON_BUTTON_BACKGROUND_COLOR,
fg=MODE_ON_BUTTON_FOREGROUND_COLOR)
            self.master.title(MODE_ON_TITLE)
            self.operating = True
            self.start_client()

```

```

    def start_client(self): # creating all mechanisms relevant, does
necessary changes to system
        self.client = VPNClient.VPNClient()
        self.client.onset_routing()
        self.client.connect(SERVER_INFO)
        self.m = client_mechanism.ClientMechanism(self.client)
        self.m.start()

    def stop_client(self): # shuts down mechanisms, reverts system changes
        self.m.stop()
        self.client.termination_routing()

    def exit_protocol(self): # responsible for graphics and stopping
mechanisms after pressing the exit button
        if tkinter.messagebox.askokcancel('Are you sure you want to exit?',
                                           'By Clicking "OK" You Will Be
Disconnected from Service'):
            if self.operating:
                self.stop_client()
                self.operating = False
            self.master.destroy()

    def keybind_operation(self, event): # runs when key-bind is pressed,
alternative to the OPERATION button
        self.operation()

```

## רפלקציה אישית:

חשבתי על הרעיון לפרויקט עוד משנה שעברה, גיששתי, חקרתי בגלל שהתעניינתי בנושא. למדתי את פרוטוקולי ההצפנות עוד לפני שלמדנו אותם בכיתה, ועברתי באופן יסודי על פרוטוקולי הרשת. נתקלתי בהרבה מאוד קשיים במהלך העבודה, סביבת העבודה שבחרתי – לינוקס, מאוד קשה לעיכול ולקח לי הרבה מאוד זמן להתרגל ולבנות את סביבת העבודה שלי, הדבר הזה לקח לי זמן רב ומאוד הטריד אותי, פחדתי ש"אתקע" ולא אצליח להתקדם, אבל בהסתכלות אחורה – בפתרון הבעיות למדתי כמה כלים וטכניקות שלא למדתי בשום מקום אחר ואני בטוח שיעזרו לי בעתיד. הדפוס הזה של בעיות שנראות כ"סוף העולם" קרה לי מספר פעמים, בכל פעם השקעתי שעות על גבי שעות כדי לדלות מידע שיסביר לי מה אני עושה ומה הבעיה, ואת האמת שזה היה מייאש, בשלב מסוים נמנעתי לחלוטין מלעבוד על הפרויקט כי לא רציתי עוד פעם להרגיש ככה תקוע. הרבה זמן שהקוד לא זז, הבנתי שאני לא יכול להתעסק רק בפתרון בעיה אחת התחלתי לעבוד מסביב, בניתי את מה שבניתי עד לאותה נקודה, הוספתי הצפנה, הוספתי גרפיקה, הוספתי ריבוי-תהליכים. למדתי הרבה, המון דברים בכתיבת הפרויקט אבל אני חושב שהייתי צריך ללכת בדרך אחרת, להשתמש ברכיב מוכן כמו אופן-וי-פי-אן, או לבחור פרויקט אחר, כי הפרויקט היה קצת גדול עליי ולא הצלחתי לסיים אותו כמו שרציתי. אני חושב שאם הייתי עובד עם חבר צוות נוסף העבודה הייתי יותר רגועה כי הייתי מקבל חוות דעת שניה לגבי המצב, והיה לי יותר מוטיבציה, במיוחד אם הייתי עובד עם חבר טוב.

לסיכום, למדתי הרבה מהפרויקט והכלים שקיבלתי אינם מבוטלים כלל אולם אם הייתה ניתנת לי ההזדמנות ללכת אחורה בזמן הייתי בוחר פרויקט אחר.

## ביבליוגרפיה

- 
- <sup>1</sup> Russo, V "LucidProgramming" (2018). "[Multiprocessing in Python: Locks](https://www.youtube.com/watch?v=...)". [www.youtube.com](https://www.youtube.com).
  - <sup>2</sup> Paulo, R (2020). "[tun -- tunnel software network interface](https://man.netbsd.org/tun)". [man.netbsd.org](https://man.netbsd.org).
  - <sup>3</sup> John, G (2019). "[What is network interface card \(NIC\)?](https://www.tutorialspoint.com/what-is-network-interface-card-nic/)". [www.tutorialspoint.com](https://www.tutorialspoint.com).
  - <sup>4</sup> Van Kempen, F, Cox, A, Blundell, P, Kleen, A. "[ifconfig](https://linux.die.net/~vankempen/ifconfig/)". [linux.die.net](https://linux.die.net).
  - <sup>5</sup> [Python os.open\(\) Method](https://www.tutorialspoint.com/python/os/open_method/). [www.tutorialspoint.com](https://www.tutorialspoint.com).
  - <sup>6</sup> Florian, T (2002). revision of document by Krasnyansky, M. "[Universal TUN/TAP device driver](https://www.kernel.org/doc/Documentation/networking/tuntap.txt)". [www.kernel.org](https://www.kernel.org).  
שורות 85-88
  - <sup>7</sup> Wikipedia user "Ikluft" (2019). "[TUN\TAP in the network stack](https://en.wikipedia.org/wiki/TUN/TAP_in_the_network_stack)". [en.wikipedia.org](https://en.wikipedia.org).
  - <sup>8</sup> "[fcntl – The fcntl and ioctl system calls](https://docs.python.org/3/library/fcntl.html)". [docs.python.org](https://docs.python.org).
  - <sup>9</sup> Florian, T (2002). revision of document by Krasnyansky, M. [Universal TUN/TAP device driver](https://www.kernel.org/doc/Documentation/networking/tuntap.txt). [www.kernel.org](https://www.kernel.org).  
שורות 67-70
  - <sup>10</sup> Both, D (2016). "[An introduction to Linux network routing](https://opensource.com/article/16/1/linux-network-routing)". [opensource.com](https://opensource.com).
  - <sup>11</sup> answer by user - v2r (2014). "[What does "chmod +x <filename>" do and how do I use it?](https://askubuntu.com/questions/74444/what-does-chmod-x-filename-do-and-how-do-i-use-it/)". [askubuntu.com](https://askubuntu.com).
  - <sup>12</sup> Reys, G (2014). "[How To Use Visudo](https://www.unixtutorial.org/how-to-use-visudo/)". [www.unixtutorial.org](https://www.unixtutorial.org).
  - <sup>13</sup> "[AES](https://he.wikipedia.org/wiki/AES)", [he.wikipedia.org](https://he.wikipedia.org).
  - <sup>14</sup> Patel, D "codebasics" (2016). "[Python Tutorial - 28. Sharing Data Between Processes Using Array and Value](https://www.youtube.com/watch?v=...)". [www.youtube.com](https://www.youtube.com)
  - <sup>15</sup> [ubuntu 20.04](https://ubuntu.com/download) download site
  - <sup>16</sup> [Python 3.8](https://python.org/download) download site
  - <sup>17</sup> [Tkinter](https://tkinter.com) – documentation
  - <sup>18</sup> [RSA library](https://pypi.org/project/cryptography/)
  - <sup>19</sup> [OS library](https://pypi.org/project/os/) (native Python library) – information site
  - <sup>20</sup> [PyCryptodome](https://pypi.org/project/pycryptodome/) library - documentation
  - <sup>21</sup> [Pickle](https://pypi.org/project/pickle/) Library (native Python library) - documentation