DIY - בניית שלט WiFi למצלמה Xiaomi-Yi

מאת Burek

הקדמה

אני מתכנן לטוס בקרוב לחופשת סנובורד ומתכוון לצלם כמה סרטונים במצלמת הספורט Xiaomi-Yi. היא נראת כך:

תפעול סטנדרטי של המצלמה דורש שימוש באפליקציה שלה. האפליקציה לא רעה אבל יש לה חסרון בולט: צריך פלאפון כדי להשתמש בה. מאחר ואני מעדיף שלא לתפעל פלאפון בזמן הגלישה, חשבתי לבנות שלט קטן למצלמה שיהיה כל כך פשוט שגם אם יהרס, אוכל לייצר אחד חדש בקלות.

כדי לייצר שלט כזה נצטרך:

1. לחקור את פרוטוקול התקשורת מול המצלמה.
2. לכתוב סקריפט לשליחת פקודות למצלמה.
3. לגרום לסקריפט לרוץ על בקר כלשהו.

מחקר הפרוטוקול

כדי להתחיל במחקר נצטרך להשיג דוגמאות (PCAP-ים) של הפרוטוקול בין הפלאפון למצלמה. הדרך הנפוצה ביותר שאני מכיר להסנפת פקטות ושמירתן לקובץ היא תוכנה בשם tcpdump. על מנת להתקין tcpdump בפלאפון שלי אצטרך לעשות לו root וזו פעולה שאני מעדיף להמנע ממנה, אז נאלצתי למצוא שיטה אחרת.

ביצוע ARP-Poisoning נראת כמו פתרון פשוט שיעשה את העבודה.

מה זה ARP-poisoning?

מויקיפדיה: "ARP הוא פרוטוקול תקשורת המשמש ברשת מחשבים לאיתור כתובת ה־MAC של תחנה ברשת על פי כתובת ה־IP שלה". כאשר רכיב ברשת מקבל את כתובת ה-MAC של תחנה אחרת ברשת הוא שומר אותה אצלו בטבלה. הרעלת ARP משמעותה לגרום באופן מכוון לטעויות בטבלאות של תחנות ברשת כדי לכוון מחדש את תעבורת הרשת כרצוננו.

אנחנו הולכים לחבר את הלפ-טופ לאותה רשת ה-Wifi שאליה מחוברים הפלאפון והמצלמה. באמצעות הרעלת ARP נגרום למצלמה להכיל את כתובת ה-MAC של הלפטופ ברשומת ה-IP של הפלאפון ונגרום לפלאפון להכיל את כתובת ה-MAC של הלפ-טופ ברשומת ה-IP של המצלמה.

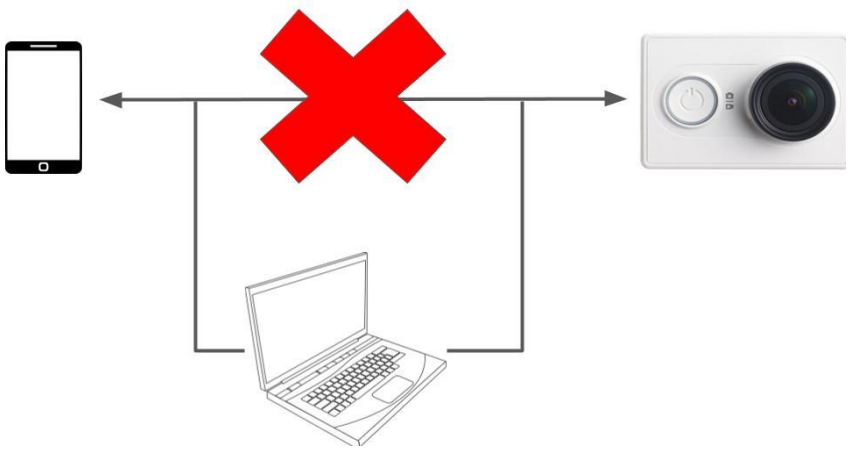
רשומות ARP לפני ההרעלה:

- בפלאפון: [<IP של המצלמה>, <MAC של המצלמה>]
- במצלמה: [<IP של הפלאפון>, <MAC של הפלאפון>]

רשומות Arp אחרי ההרעלה:

- בפלאפון: [<IP של המצלמה>, <MAC של הלפ-טופ>]
- במצלמה: [<IP של הפלאפון>, <MAC של הלפ-טופ>]

או בצורה ויז'ואלית:

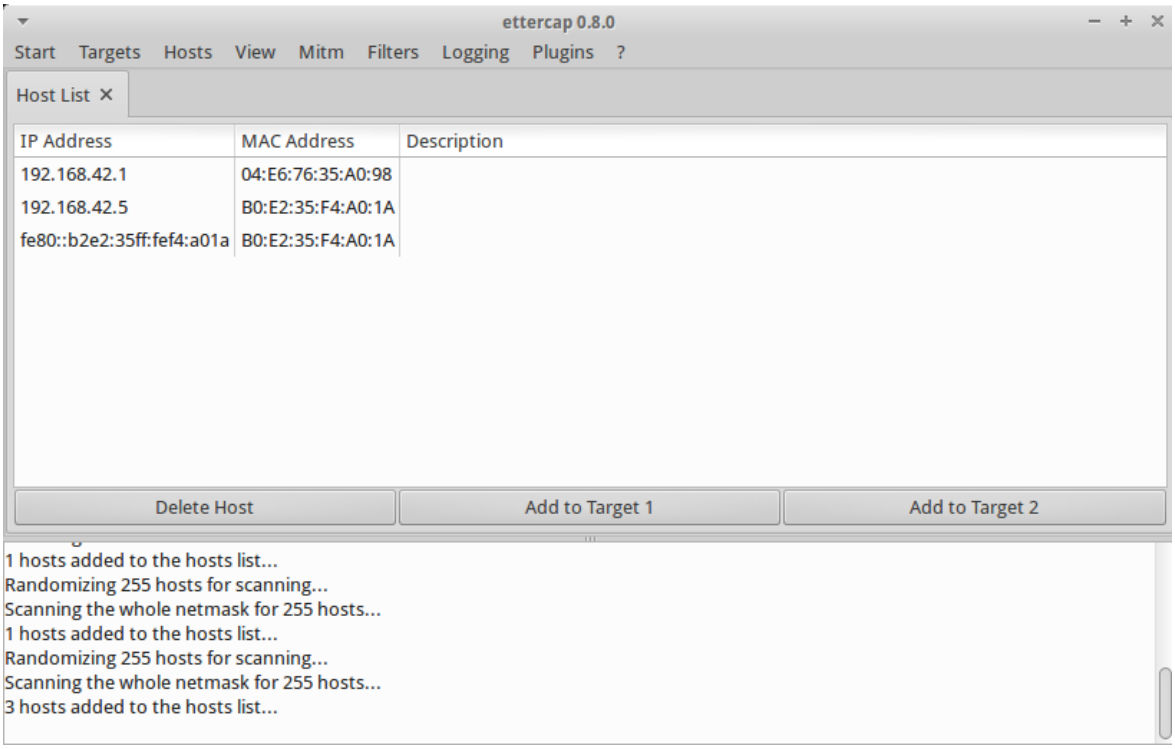


ביצוע ARP-Poisoning

כדי להפעיל את ההתקפה הזו על המצלמה והפלאפון אנחנו הולכים להשתמש בתוכנה בשם Ettercap. הפעלת Arp-poisoning ב-Ettercap היא מאוד פשוטה, כל מה שצריכים לעשות זה לבחור שתי מטרות ולהפעיל את ההתקפה:

בחירת מטרות:

- נפתח את תפריט Hosts ונבחר ב-"host list".
- נלחץ Ctrl-S כדי לסרוק את הHosts ברשת - אנחנו מצפים לראות רק את המצלמה והפלאפון.
- נבחר מטרות על ידי לחיצה על Add to target 1/2



הפעלת ההרעלה: מתוך התפריט Mitm נבחר Arp poisoning ונסמן Sniff remote connections.

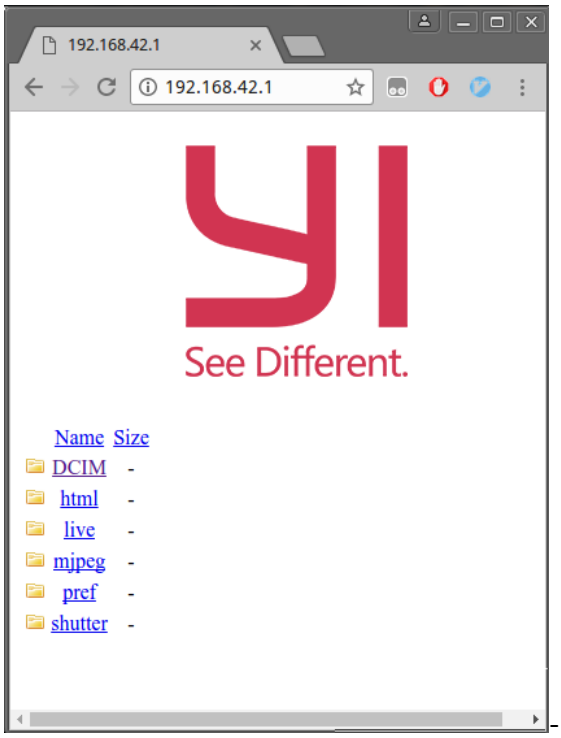
בשלב זה התעבורה בין הפלאפון למצלמה עוברת דרכינו. תוכנה שאני אוהב להשתמש בה להפרדת TCP-sessions היא tcpflow. דוגמאת הרצה של tcpflow על ממשק ה-Wifi:

```
b@b-X370:~/Programs/Flows$ sudo tcpflow -i wlan0
tcpflow: listening on wlan0
^Ctcpflow: terminating
b@b-X370:~/Programs/Flows$ ls
192.168.042.001.00554-192.168.042.005.49042  192.168.042.005.34548-
192.168.042.001.07878  alerts.txt
192.168.042.001.07878-192.168.042.005.34548  192.168.042.005.49042-
192.168.042.001.00554  report.xml
```

ניתוח הפרוטוקול

כעת, משקיבלנו הסנפות של הפרוטוקול אפשר להתחיל לחקור אותו ולראות אם אפשר לחקות אותו מחוץ לאפליקציה הייעודית. מהסתכלות מהירה על החיבורים בין הפלאפון למצלמה אפשר לזהות שלושה ממשקים (כל אחד על port משלו):

- פורט 7878 - ערוץ פיקוד מבוסס JSON. מכיל פקודות כמו שינוי קונפיגורציה וצילום. (מעניין)
- פורט 80 - ממשק דפדפן למצלמה (קצת מעניין)



- פורט 554 - הזרמת וידאו חי מהמצלמה לפלאפון (לא מעניין)

כשבוחנים את המידע שעובר על ערוץ הפיקוד, אפשר לזהות שמטרת הבקשה הראשונה היא קבלת מזהה Session token אותו נצטרך לשלוח למצלמה בכל פעם שנרצה לבקש ממנה לבצע פעולה:

```
** session token request **
{"msg_id":257,"token":0,"heartbeat":1,"param":0}
{ "msg_id": 7, "type": "vf_stop" }
{ "rval": 0, "msg_id": 257, "param": 2 }

{"msg_id":3,"token":2}
…
```

בזמן שעבדתי על מיפוי שאר הפקודות מצאתי שמישהו אחר כבר עשה את המיפוי:

https://gist.github.com/SkewPL/f57e6cff7fa14601f6b256926aa33437

(תזכורת לכך שתמיד כדאי לחפש בגוגל. גם אם חשבתי לתומי שאני היחיד שחוקר את המצלמה...)

מימוש סקריפט פיתון לשליטה במצלמה

אז אנחנו חושבים שאנחנו יודעים איך לשלוט במצלמה. זה לא מספיק! נצטרך ליישם זאת על ידי כתיבת סקריפט שמממש את הפרוטוקול ולאחר מכן נבדוק אותו מול המצלמה.

הסקריפט מאוד טריוויאלי ועושה את המינימום האפשרי בשביל לתקשר עם המצלמה (כמיטב מסורת הוכחות ההתכנות - הוא מתעלם לחלוטין משגיאות שעלולות לקרות).

```
import json
import socket
import pprint

IP = "192.168.42.1"
PORT = 7878

class CamHandler(object):
    def __init__(self, ip, port):
        self.sock = socket.socket()
        self.sock.connect((ip,port))
        self.token = 0

    def _get_token(self):

self.sock.send('{"msg_id":257,"token":0,"heartbeat":1,"param":0}')
        data = ""
        while True:
            data = self.sock.recv(10000)
            print "Data:", data
            res = json.loads(data)
            print "Parsed:", res
            if res['msg_id'] == 257:
                self.token = res['param']
                print "TOKEN:", self.token
                break

    def do_command(self, command):
        command['token'] = self.token
        self.sock.send(json.dumps(command))
        while True:
            res = json.loads(self.sock.recv(10000))
            pprint.pprint(res)
            if res['msg_id'] == command['msg_id']:
                break

    def get_battery(self):
        command = {"msg_id":13}
        self.do_command(command)

    def take_picture(self):
        command = {"msg_id":16777220,"token":5,"param":"precise
quality;off"}
        self.do_command(command)

    def start_recording(self):
        command = {"msg_id":513,"token":5}
        self.do_command(command)
```

```
    def stop_recording(self):
        command = {"msg_id":514,"token":5}
        self.do_command(command)

    def get_camera_params(self):
        command = {"msg_id":3,"token":5}
        self.do_command(command)

    def set_param(self, param, value):
        command = {"msg_id":2,
                   "type":param,
                   "param":value,
                   "token":1}
        self.do_command(command)

    def test(self):
        command = {"msg_id":260,"token":5}
        self.do_command(command)

def main():
    ch = CamHandler(IP, PORT)
    ch._get_token()
    ch.get_camera_params()
    ch.take_picture()


if __name__ == "__main__":
    main()
```

תאלצו לסמוך עלי שהסקריפט עובד. כמובן שאתם מוזמנים לבדוק בעצמכם. עדיין לא סיימנו. הסקריפט הזה רץ על מחשב ואין סיכוי שאחזיק לפ-טופ בזמן שאני גולש...

הרצת הקוד על ESP8266

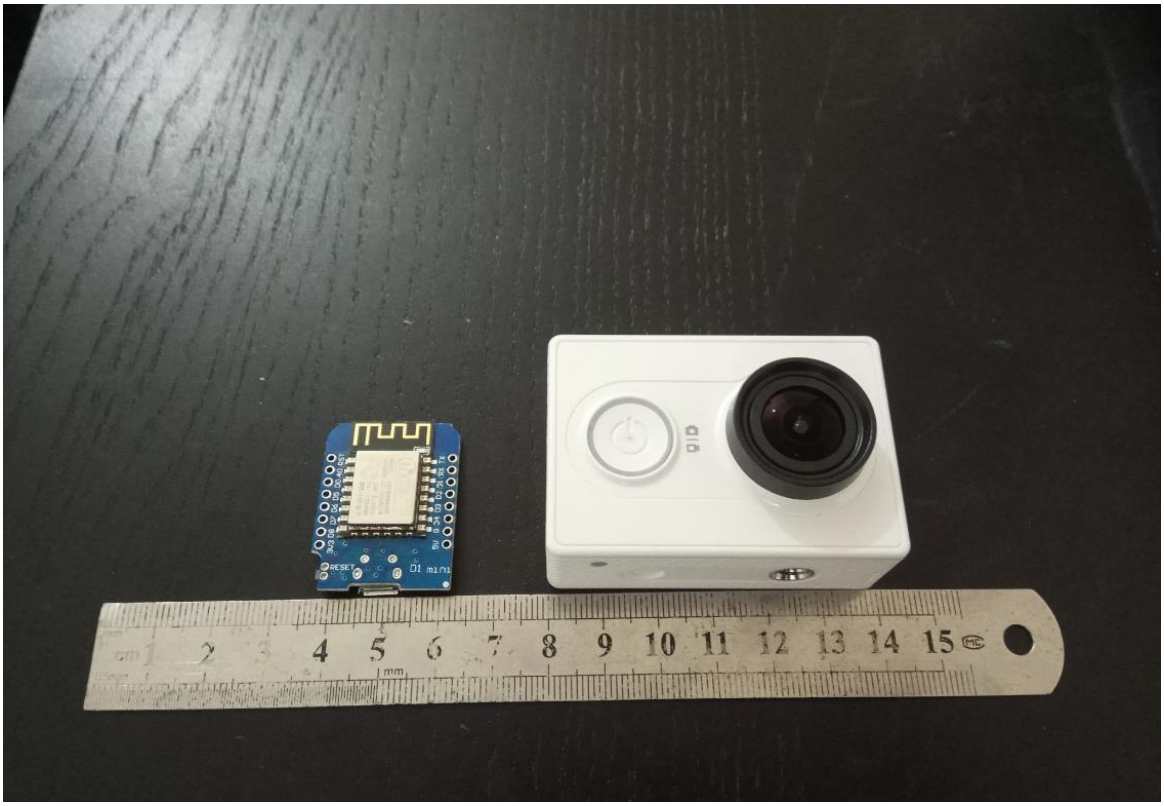
מה זה ESP8266?

מדובר על בקר שלדעתי הוא לא פחות מהסכין השוויצרית של בקרים שתומכים ב-Wifi. הוא מאוד זול (כ-12 ש"ח כולל משלוח), יש קהילה מעולה שתומכת בו ואפשר לשאול בה שאלות, והדובדבן שבקצפת - יש גרסת MicroPython שניתן להריץ עליו.

אם אלו ימיכם הראשונים בעולם הבקרים, אני ממליץ לקנות את ה-Wemos D1 mini. זה בורד שמכיל את ה-ESP8266 וגם ממשק usb שמאפשר להעלות קוד לבקר בצורה מאוד נוחה.

מה זה MicroPython?

מימוש של ה-Interpreter של פיתון 3 המיועד לבקרים. זה מקל משמעותית על תהליך הפיתוח כי אפשר להנות מהאבסטרקציות הנוחות של פיתון למרות שעובדים על בקר. כמובן שאנחנו משלמים על ההנאה הזו בכוח עיבוד, אבל במקרה שלנו זה דיי זניח.



לפני שנוכל להריץ את הסקריפט על הבקר נצטרך להעלות אליו את Micropython:

```
sudo esptool.py --port /dev/ttyUSB0 erase_flash
sudo esptool.py --port /dev/ttyUSB0 write_flash -fm dio 0x00000 esp8266-20171101-micropython-v1.9.3.bin
```

נצטרך גם להוסיף קוד שמתחבר לרשת ה-Wifi של המצלמה:

```
import network
station = network.WLAN(network.STA_IF)
station.active(True)
station.scan()
station.connect("CameraWifi", "WifiPassword")
station.ifconfig()
```

נרצה להריץ את הסקריפט בעליה של הבקר ולכן נכתוב אותו בקובץ boot.py. כן, micropython מגיע עם מערכת קבצים!

סקריפט מוכן להרצה:

```
import network
import time

station = network.WLAN(network.STA_IF)
station.active(True)
SSID = "YDXJ_----"
PASSWORD = "password"

station.connect(SSID, PASSWORD)
while True:
     current_ip = station.ifconfig()[0]
     if current_ip != "0.0.0.0":
          break
     print ("Waiting for ip")
     time.sleep(1)

import json
import socket

IP = "192.168.42.1"
PORT = 7878

class CamHandler(object):
    def __init__(self, ip, port):
        self.sock = socket.socket()
        self.sock.connect((ip,port))
        self.token = 0

    def _get_token(self):

self.sock.send('{"msg_id":257,"token":0,"heartbeat":1,"param":0}')
        data = ""
        while True:
            data = self.sock.recv(10000)
            print ("Data:", data)
            res = json.loads(data)
            print ("Parsed:", res)
            if res['msg_id'] == 257:
                self.token = res['param']
                print ("TOKEN:", self.token)
                break

    def do_command(self, command):
        command['token'] = self.token
```

```
        self.sock.send(json.dumps(command))
        while True:
            res = json.loads(self.sock.recv(10000))
            print(res)
            if res['msg_id'] == command['msg_id']:
                break

    def get_battery(self):
        command = {"msg_id":13}
        self.do_command(command)

    def take_picture(self):
        command = {"msg_id":16777220,"token":5,"param":"precise
quality;off"}
        self.do_command(command)

    def start_recording(self):
        command = {"msg_id":513,"token":5}
        self.do_command(command)

    def stop_recording(self):
        command = {"msg_id":514,"token":5}
        self.do_command(command)

    def get_camera_params(self):
        command = {"msg_id":3,"token":5}
        self.do_command(command)

    def set_param(self, param, value):
        command = {"msg_id":2,
                   "type":param,
                   "param":value,
                   "token":1}
        self.do_command(command)

    def test(self):
        command = {"msg_id":260,"token":5}
        self.do_command(command)

def main():
    ch = CamHandler(IP, PORT)
    ch._get_token()
    ch.take_picture()

main()
```

סיימנו - יש לנו צ'יפ שמתחבר אל המצלמה וגורם לה לצלם תמונה.

כדי לעשות שימוש אמיתי בבקר נצטרך להלחים אליו כמה כפתורים ולייצר לו קופסה כלשהי אבל זה כבר לפעם הבאה. ☺

סיכום

במאמר זה עברנו על כמה שיטות המאפשרות לחקור פרוטוקולים בצורה מאוד פשוטה וקלה גם אם אין לנו יכולת להריץ קוד על הרכיבים שאנחנו רוצים לחקור. בנוסף, יש במאמר גם טעימה מעולם הבקרים ואולי זה אפילו יהווה כניסה קלה לאנשים שרוצים להתחיל להתעסק בבקרים ולא כל כך יודעים איפה להתחיל.

על המחבר

בעברי ביצעתי מגוון תפקידים בחיל המודיעין. כיום עובד בסייבריזון כמפתח בתחום ה--Windows Internals ולא מזמן פתחתי בלוג טכנולוגי: burek.tech, מוזמנים להתרשם.

אם יש שאלות או תיקונים שהייתם רוצים במאמר מוזמנים לשלוח ל-burektech@gmail.com.

תודה,
Burek

קישורים

ויקיפדיה על ARP:

https://he.wikipedia.org/wiki/Address_Resolution_Protocol

ויקיפדיה על ARP Poisoning:

https://he.wikipedia.org/wiki/ARP_spoofing

Ettercap:

https://ettercap.github.io/ettercap/about.html

MicroPython:

https://docs.micropython.org/en/latest/esp8266/

ESPTool:

https://github.com/espressif/esptool