Reverse Engineering the Eufy Ecosystem: a Deep Dive Into Security Vulnerabilities and Proprietary Protocols

Victor Goeman, Dairo de Ruck, Tom Cordemans, Jorn Lapon, Vincent Naessens

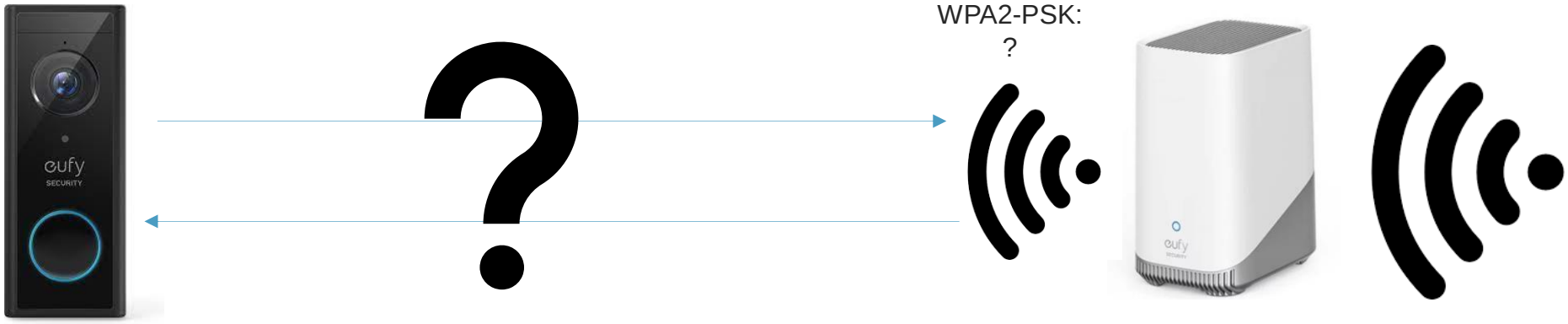
DistriNet

Who is Eufy?

Eufy

- Founded in 2016
- Already among market leaders
- Focus on security
- 1/334 Households

Eufy – Setup

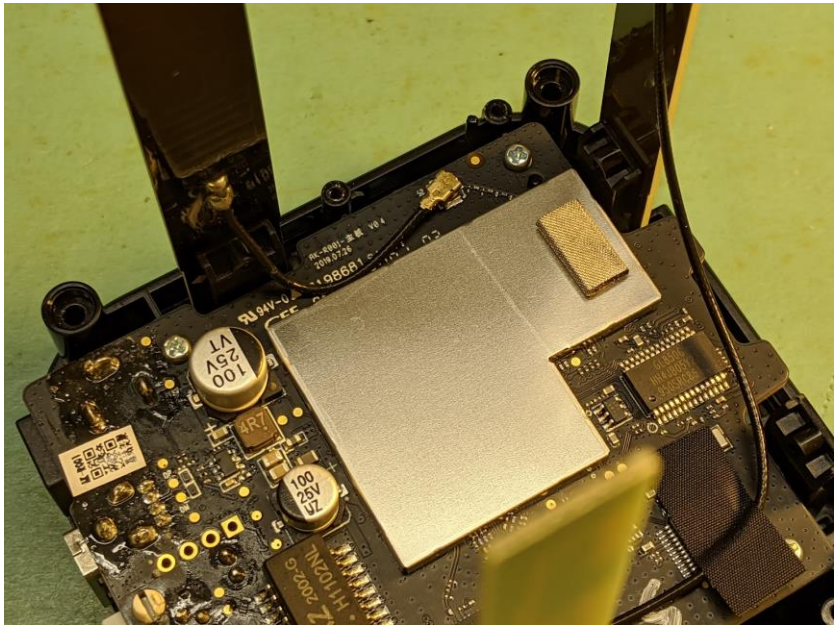


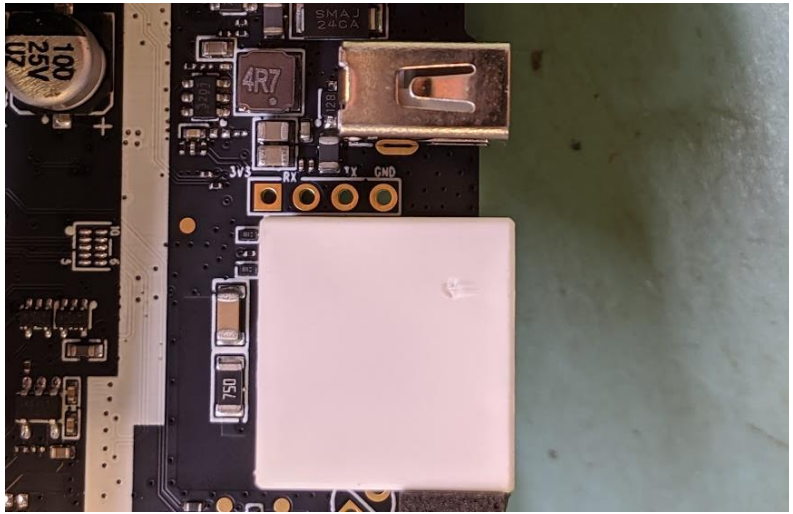
Methods

1. Network traffic analysis
2. Firmware analysis
3. Symbolic analysis

Cracking the WPA2-PSK

Eufy – Firmware acquisition





Eufy – Firmware acquisition

```
  ** send data to uart2 **
FE 01 21 58 00 78


 ++ read from uart2 data:
FE 02 61 58 00 10 2B

 success to get respance.
get response from app. code:0x00.
in normal mode.
  ** send data to uart2 **
FE 01 21 51 00 71


 ++ read from uart2 data:
FE 01 61 51 00 31

 success to get respance.


3: System Boot system code via Flash.
## Booting image at bc050000 ...
raspi_read: from:50000 len:40
   Image Name:   Linux Kernel Image
   Image Type:   MIPS Linux Kernel Image (lzma compressed)
   Data Size:    14616169 Bytes = 13.9 MB
   Load Address: 80000000
   Entry Point:  805e9490
raspi_read: from:50040 len:2906a9
```

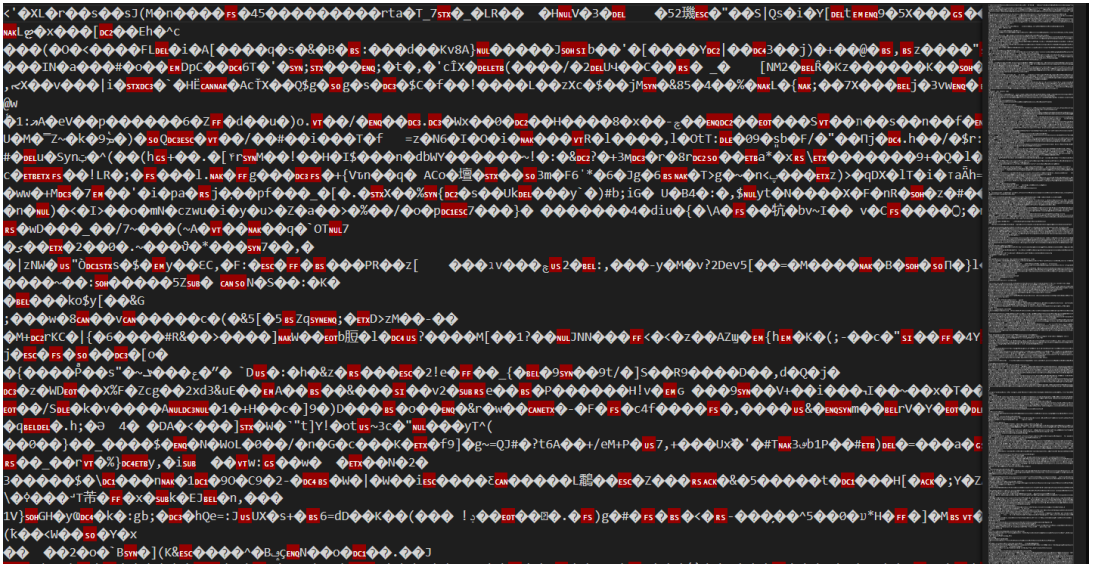
```
normal.sh current_year:1970
normal.sh fail to get system time. ignore ...
[   50.488000] ####Set_SignalUserPid_Proc,5897

Login timed out process '/bin/login' (pid 5290) exited. Scheduling for restart.
setrlimit(RLIMIT_CORE, &limit);
starting pid 6382, tty '/dev/ttyS1': '/bin/login'
Eufycamera login:
```

```
 ##### The CPU freq = 575 MHZ ####
 estimate memory size =128 Mbytes
RESET MT7628 PHY!!!!!!
Please choose the operation:
   1: Load system code to SDRAM via TFTP.
   2: Load system code then write to Flash via TFTP.
   3: Boot system code via Flash (default).
   4: Entr boot command line interface.
   7: Load Boot Loader code then write to Flash via Serial.
   9: Load Boot Loader code then write to Flash via TFTP.

   m: Load mini system code then write to Flash via TFTP.
   u: enter mini system for upgrade normal system.
```

DistriNet

Eufy – Firmware acquisition

- Copying the MTD blocks




```
   4096 Feb 20 08:14 bin
   4096 Dec 31  1969 dev
   4096 May  9 08:41 etc
   4096 Feb 20 08:14 etc_ro
4349716 Apr 26 08:17 home_security
   4096 May 10 09:24 lib
   4096 May  9 09:35 media
   4096 Dec 31  1969 mnt
  23664 Apr 27 08:29 ntpclient
   4096 Dec 31  1969 proc
   4096 Feb 20 08:14 sbin
   4096 Dec 31  1969 sys
   4096 May  9 09:38 tmp
   4096 Feb 20 08:13 usr
   4096 May  9 09:37 var
6669272 Apr 27 08:30 webrtc_client
```

Eufy – Authentication bypass - CVE-2023-37822

Unencrypted

WPA2-PSK:
“YzQ3MzI3”

Serial number:
“T8010P23224107B0”

DistriNet

Eufy – Filesystem analysis

- Suspicious activities
  - Reuse of password
    - UART login
    - WPA2-PSK
  - Password follows a pattern
    - Not random
- Dive deeper!

```
genSysFiles()
    #login=`nvram_get 2860 Login`
    #pass=`nvram_get 2860 Password`
    login="eufycamera"
    pass="`nvram_get WPAPSK1`"
    if [ "$pass" == "" ];then
        pass="Anker39#*"
    #login="admin"
    #pass="admin"
    if [ "$login" != "" -a "$pass" != "" ]; then
    echo "$login::0:0:Adminstrator:/:/bin/sh" > /etc/passwd
    echo "$login:x:0:$login" > /etc/group
        chpasswd.sh $login $pass
    if [ "$CONFIG_PPPOL2TP" == "y" ]; then
    echo "l2tp 1701/tcp l2f" > /etc/services
    echo "l2tp 1701/udp l2f" >> /etc/services
configVIF()
```

Eufy – Filesystem analysis

```
[ERROR][log.c:LOG_vprintf:463] - appClientInit OK
[ERROR][log.c:LOG_vprintf:463] - appClientInit OK, create main thread=0x22c6398!
[INFO][sub1g_interface.c:zx_init_sub1g:4952] - init_sub1g enter,use /etc_ro/appclient.cfg. ret=0
[INFO][sys_interface.c:zx_create_thread:1128] - stacksize = 262144 Byte
[INFO][sys_interface.c:zx_create_thread:1166] - tid:28701, thread_type:1 ok
[INFO][security_main.c:zx_watch_dog:1760] - zx_watch_dog, PID=5839----------
[INFO][security_main.c:zx_watch_dog:1772] - gRecord_pipe:11,12
[INFO][sub1g_interface.c:zx_get_gw_info_cnf:589] - Get Nwk Info Confirmed, Status=1 ,collector extAddr = 0x417b6659c5f,panID = 0x9c5f
[INFO][sub1g_interface.c:zx_get_gw_info_cnf:600] - using anker_mac: 0x4:17
[INFO][remote_system.c:zx_remote_system_accept_node:273] - cmd:/sbin/chpasswd.sh eufycamera YzQ3MzI3 need_response:0
[INFO][remote_system.c:zx_remote_system_cmd:104] - 000 alloc send cmdbuf:/sbin/chpasswd.sh eufycamera YzQ3MzI3, need_response:0, node:0x22d3db8
[INFO][xm_interface.c:zx_init_camera_sdk:5314] - enter......
[INFO][remote_system.c:zx_remote_system_send_server:178] - get socket:13
[INFO][remote_system.c:zx_remote_system_free_node:367] - 222 find delete node:0x22d3db8, cmdbuf:/sbin/chpasswd.sh eufycamera YzQ3MzI3, fd:13
[INFO][xm_interface.c:zx_init_camera_sdk:5358] - XMLIB VERSION: V-3.0.8.8  <===========
[INFO][xm_interface.c:zx_init_camera_sdk:5419] - xmsdk:0, mac:04:17:B6:66:45:C0, status:1, IP:192.168.32.5
[INFO][xm_interface.c:zx_init_camera_sdk:5419] - xmsdk:1, mac:, status:0, IP:
[INFO][xm_interface.c:zx_init_camera_sdk:5419] - xmsdk:2, mac:, status:0, IP:
[INFO][xm_interface.c:zx_init_camera_sdk:5419] - xmsdk:3, mac:, status:0, IP:
[INFO][xm_interface.c:zx_init_camera_sdk:5419] - xmsdk:4, mac:, status:0, IP:
[INFO][xm_interface.c:zx_init_camera_sdk:5419] - xmsdk:5, mac:, status:0, IP:
[INFO][xm_interface.c:zx_init_camera_sdk:5419] - xmsdk:6, mac:, status:0, IP:
[INFO][xm_interface.c:zx_init_camera_sdk:5419] - xmsdk:7, mac:, status:0, IP:
```

Eufy – Reverse engineering

DistriNet

Eufy – Reverse engineering

_securityClean

```
00784d70 53 53 49        ds          "SSID2"
         44 32 00
00784d76 00              ??          00h
00784d77 00              ??          00h

                    s_ral_SSID=%s_00784d78                          XREF[2]:     get_wifi_ral_info_v0:00513820(*),
                                                                                 get_wifi_ral_info:00513aa8(*)
00784d78 72 61 31        ds          "ral SSID=%s"
         20 53 53
         49 44 3d ...

                    s_WPAPSK2_00784d84                              XREF[3]:     get_wifi_ral_info_v0:00513860(*),
                                                                                 get_wifi_ral_info:00513ae8(*),
                                                                                 create_dev_pair_wav:005142e0(*)
00784d84 57 50 41        ds          "WPAPSK2"
         50 53 4b
         32 00

                    s_ral_WIFI_PWD=%s_00784d8c                      XREF[2]:     get_wifi_ral_info_v0:005138e4(*),
                                                                                 get_wifi_ral_info:00513b6c(*)
00784d8c 72 61 31        ds          "ral WIFI_PWD=%s"
         20 57 49
         46 49 5f ...

                    s_/sbin/chpasswd.sh_eufycamera_%s_00784d9c      XREF[1]:     FUN_00513c24:00513da0(*)
00784d9c 2f 73 62        ds          "/sbin/chpasswd.sh eufycamera %s"
         69 6e 2f
         63 68 70 ...
```


DistriNet

Eufy – Reverse engineering

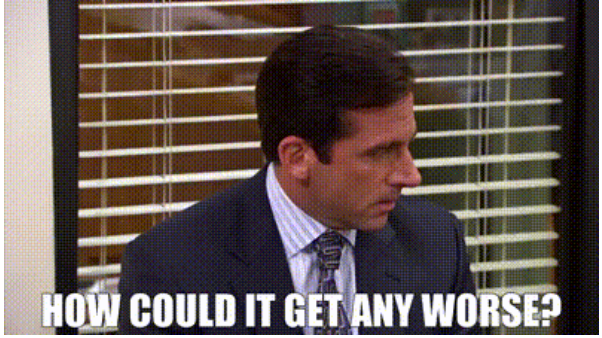
```
     undefined4        Stack[-0x3c]:4 local_3c                                XRE
     undefined4        Stack[-0x40]:4 local_40                                XRE
                     F00513C24                                       XREF[3]:

00513c24 b0 ff bd 27     addiu       sp,sp,-0x50
     assume t9 = <UNKNOWN>
     assume gp = <UNKNOWN>
00513c28 4c 00 bf af     sw          ra,local_4(sp)
00513c2c 84 00 1c 3c     lui         gp,0x84
00513c30 30 b0 9c 27     addiu       gp,gp,-0x4fd0
00513c34 20 00 bc af     sw          gp=>_gp_1,local_30(sp)
00513c38 50 00 a4 af     sw          a0,local_res0(sp)
00513c3c 50 00 a2 8f     lw          v0,local_res0(sp)
00513c40 00 00 00 00     nop
00513c44 07 00 40 10     beq         v0,zero,LAB_00513c64
00513c48 00 00 00 00     _nop
00513c4c 50 00 a2 8f     lw          v0,local_res0(sp)
00513c50 00 00 00 00     nop
00513c54 00 00 42 80     lb          v0,0x0(v0)
00513c58 00 00 00 00     nop
00513c5c 04 00 40 14     bne         v0,zero,LAB_00513c70
00513c60 00 00 00 00     _nop

                     LAB_00513c64                                    XREF[1]:
00513c64 ff ff 02 24     li          v0,-0x1
00513c68 cb 4f 14 08     j           LAB_00513f2c
00513c6c 00 00 00 00     _nop

                     LAB_00513c70                                    XREF[1]:
00513c70 28 00 a0 af     sw          zero,local_28(sp)
00513c74 2c 00 a0 af     sw          zero,local_24(sp)
00513c78 50 00 a4 8f     lw          a0,local_res0(sp)
00513c7c 38 83 82 8f     lw          v0,-0x7cc8(gp)=>->zx_MD5_HexStr
00513c80 00 00 00 00     nop
00513c84 21 c8 40 00     move        t9,v0
00513c88 09 f8 20 03     jalr        t9=>zx_MD5_HexStr
00513c8c 00 00 00 00     _nop
00513c90 20 00 bc 8f     lw          gp,local_30(sp)
00513c94 30 00 a2 af     sw          v0,local_20(sp)
00513c98 30 00 a2 8f     lw          v0,local_20(sp)
00513c9c 00 00 00 00     nop
00513ca0 a1 00 40 10     beq         v0,zero,LAB_00513f28
```

Decompile: FUN_00513C24 - (home_securityClean)

```
1
2  undefined4 FUN_00513C24(char *param_1)
3
4  {
5    char *pcVar1;
6    size_t sVar2;
7    char *__src;
8    int iVar3;
9    undefined4 local_1c;
10   undefined4 local_18;
11   undefined4 local_14;
12   undefined4 local_10;
13
14   if (((param_1 != (char *)0x0) && (*param_1 != '\0')) &&
15       (pcVar1 = (char *)zx_MD5_HexStr(param_1), pcVar1 != (char *)0x0)) {
16     sVar2 = strlen(pcVar1);
17     __src = (char *)zx_Memory_Encode(pcVar1,sVar2);
18     if (pcVar1 != (char *)0x0) {
19       free(pcVar1);
20     }
21     if (__src != (char *)0x0) {
22       local_1c = 0;
23       local_18 = 0;
24       local_14 = 0;
25       local_10 = 0;
26       strncpy((char *)&local_1c,__src,8);
27       if (__src != (char *)0x0) {
28         free(__src);
29       }
30       zx_do_system("/sbin/chpasswd.sh eufycamera %s",&local_1c);
31       nvram_init(1);
32       pcVar1 = (char *)nvram_bufget(1,"WPAPSK1");
33       if ((pcVar1 != (char *)0x0) && (iVar3 = strcmp((char *)&local_1c,pcVar1), iVar3 != 0)) {
34         nvram_bufset(1,"WPAPSK1",&local_1c);
35         nvram_commit(1);
36         nvram_close(1);
37         zx_do_system("iwpriv ra0 set WPAPSK=%s",&local_1c);
38         dzlog("src/sys_interface.c",0x13,"set_wifi_pwd",0xc,0x22f,0x28,"set WIFI_PWD successed!");
39         return 0;
40       }
41       nvram_close(1);
42     }
43   }
44   return 0xffffffff;
```

Eufy – Reverse engineering

- WPA2-PSK directly linked to serial number?!




```
int set_wifi_pwd(char *serialNumber)

{
  char *md5HashSN_hex;
  size_t length;
  char *ActualPassword;
  int iVar1;
  char *passwordPointer;

  if (((serialNumber != (char *)0x0) && (*serialNumber != '\0')) &&
     (md5HashSN_hex = (char *)zx_MD5_HexStr(serialNumber), md5HashSN_hex != (char *)0x0)) {
    length = strlen(md5HashSN_hex);
    ActualPassword = (char *)zx_Memory_Encode(md5HashSN_hex,length);
    if (md5HashSN_hex != (char *)0x0) {
      free(md5HashSN_hex);
    }
    if (ActualPassword != (char *)0x0) {
      passwordPointer = (char *)0x0;
      strncpy((char *)&passwordPointer,ActualPassword,8);
      if (ActualPassword != (char *)0x0) {
        free(ActualPassword);
      }
      zx_do_system("/sbin/chpasswd.sh eufycamera %s",&passwordPointer);
      nvram_init(1);
      md5HashSN_hex = (char *)nvram_bufget(1,"WPAPSK1");
      if ((md5HashSN_hex != (char *)0x0) &&
         (iVar1 = strcmp((char *)&passwordPointer,md5HashSN_hex), iVar1 != 0)) {
        nvram_bufset(1,"WPAPSK1",&passwordPointer);
        nvram_commit(1);
        nvram_close(1);
        zx_do_system("iwpriv ra0 set WPAPSK=%s",&passwordPointer);
        dzlog("src/sys_interface.c",0x13,"set_wifi_pwd",0xc,0x22f,0x28,"set WIFI_PWD successed!");
        return 0;
      }
      nvram_close(1);
    }
  }
  return -1;
```

Eufy – Authentication Bypass

Serial number:
T8010P23224107B0

MD5 hash

c47327

Base64

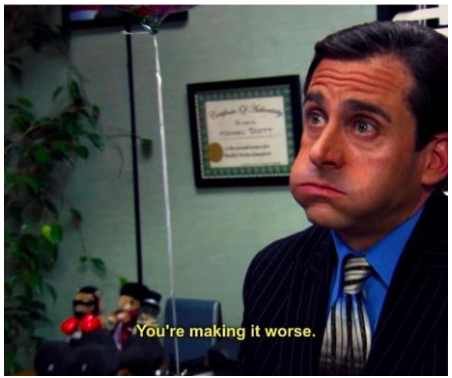
YzQ3MzI3

First 8 bytes

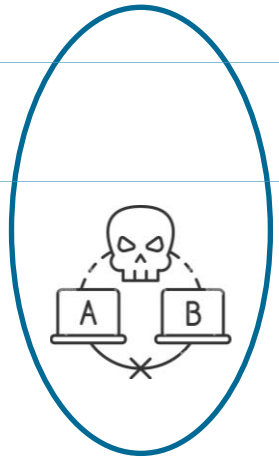
WPA2PSK:
“YzQ3MzI3”

Entropy? #possible combinations?

16^6 = 16.777.216



Eufy – Authentication bypass



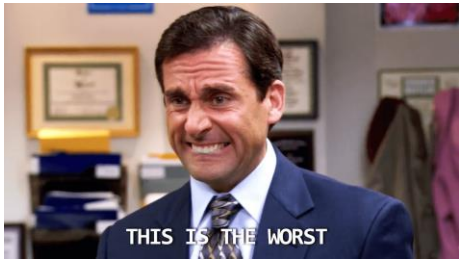
WPA2-PSK:
XXXXXXXX
“YzQ3MzI3”

Serial number:
“T8010P23224107B0”

- Steal WPA Handshake
- Crack WPA2-PSK – Brute force
  - 20 seconds

DistriNet

Authentication Bypass



WPA2-PSK:
“YzQ3MzI3”

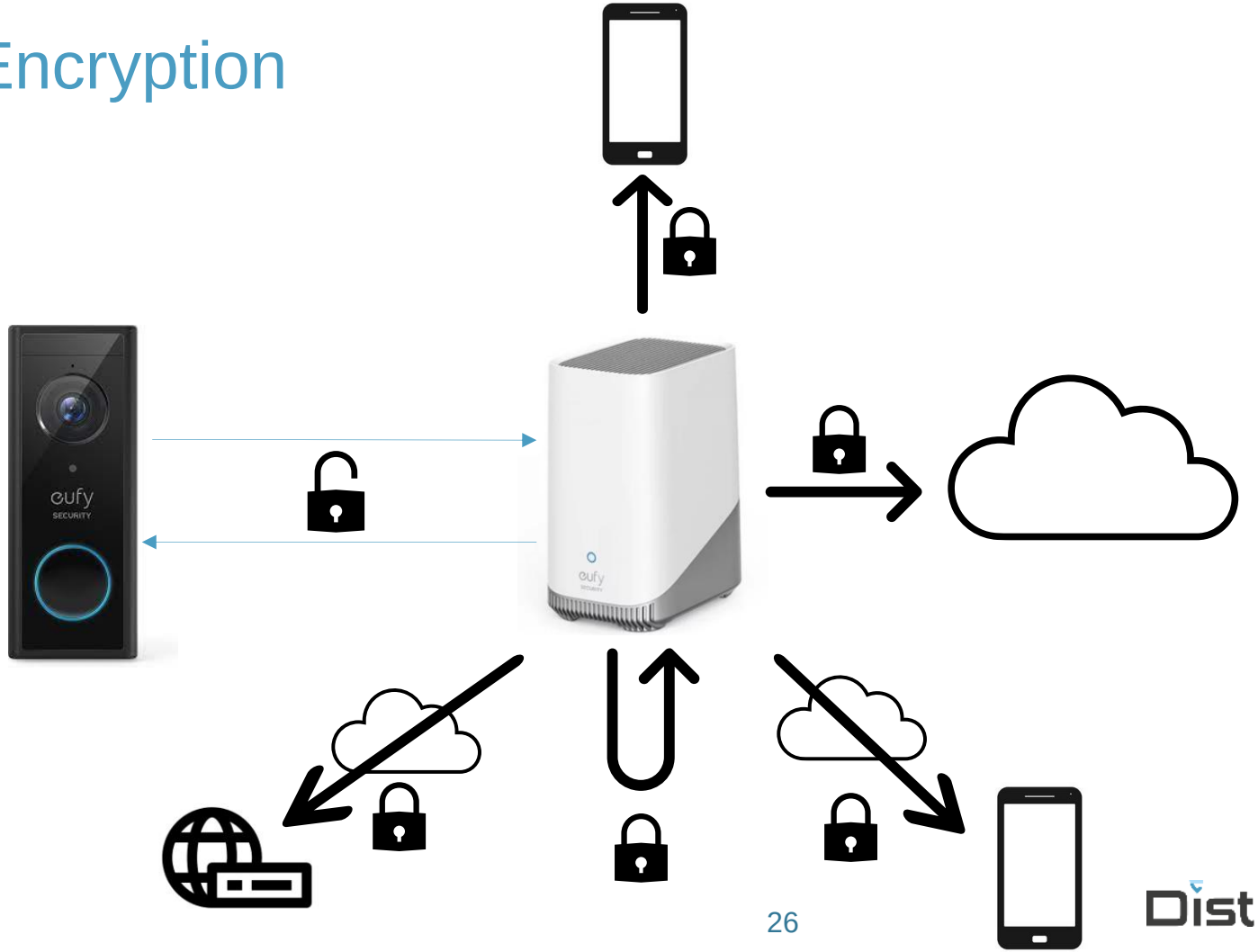
Serial number:
“T8010P23224107B0”

DistriNet

Breaking the Encryption

Eufy – Encryption

Eufy – Encryption



DistriNet

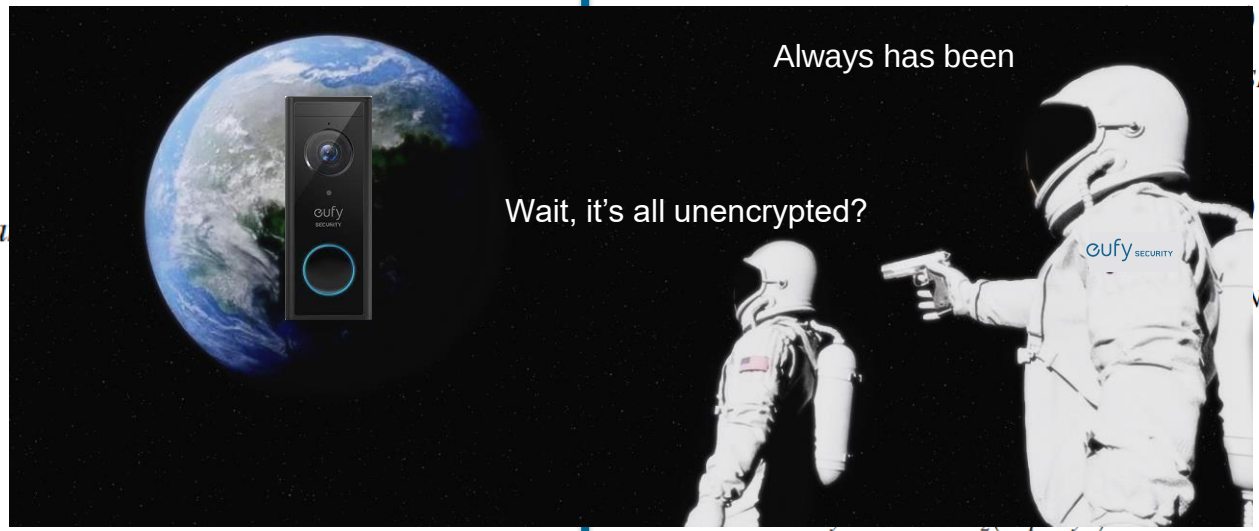
Eufy – Encryption

P2P Encryption

$Key = basePara$

Media Encryption

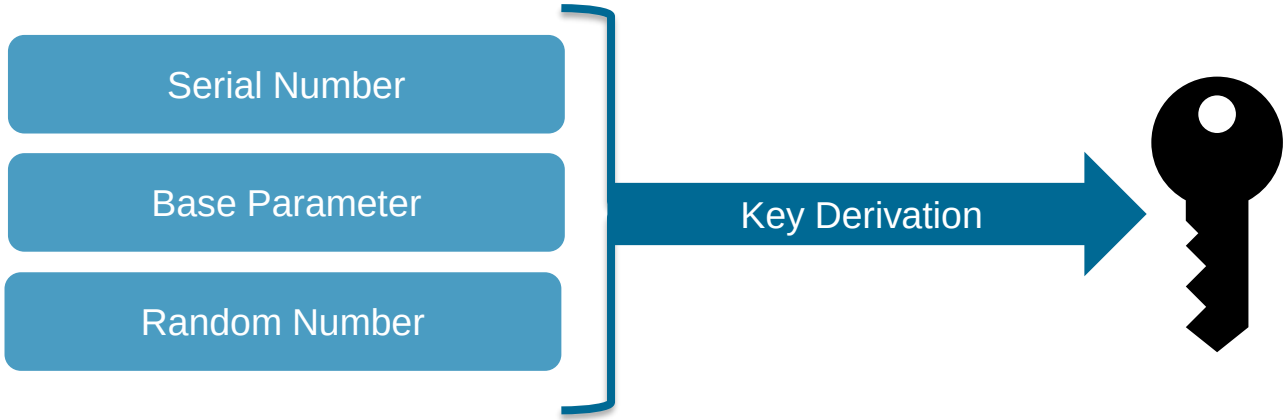
1: **function** GENPICBASECODE(*serialNumber*, *baseParam*)



18: **return** *encKey*
19: **end function**

DistriNet

Eufy - Encryption



Eufy - Encryption

Serial Number

Base Parameter

Random Number

Key Derivation

Eufy header

Encrypted JFIF header

Unencrypted JFIF body

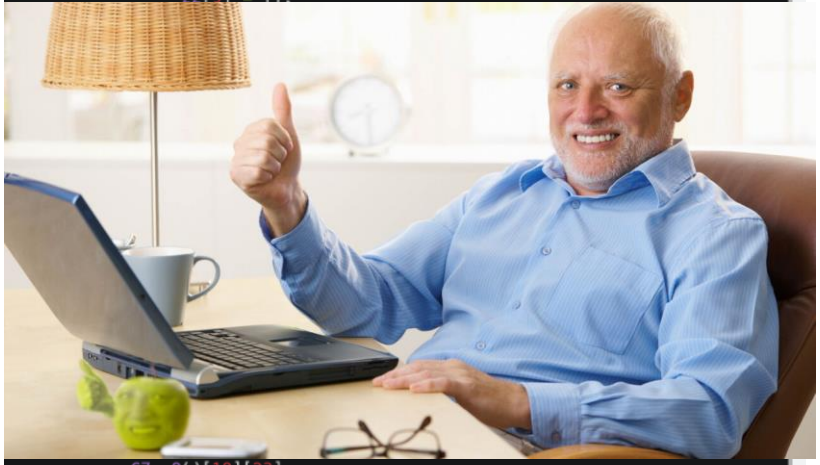
DistriNet

Eufy – Encryption

- Manual reverse engineering
  - Complex
  - Fault prone
  - Time consuming

Selective execution

```
var C7 = M7wd_;
var s2 = [arguments];
s2[6] = C7.U1()[0][5];
C7.P1();
for (; s2[6] !== C7.q8()[26][3];) {
  switch (s2[6]) {
    case C7.U1()[3][6]:
      s2[9][C7.F(4)]((function () {
        C7.P1();
        var z2 = [arguments];
        z2[6] = C7.q8()[25][8];
        for (; z2[6] !== C7.U1()[1][7];) {
          switch (z2[6]) {
            case C7.U1()[31][14]:
```



```
      case C7.q8()[18][23]:
        s2[9] = [];
        s2[6] = C7.U1()[28][19];
        break;
      case C7.U1()[8][16]:
        s2[4] = s2[0][0][s2[5]];
        s2[3] = V0oXS$(s2[4]);
        s2[6] = C7.q8()[15][18];
        break;
      case C7.U1()[21][8]:}
}
```

Eufy - Encryption

Selective execution

- Objective: Reconstruct AES key
  - Run cross-architecture binary
  - Run targeted functions
  - Run with concrete inputs
  - Avoid complex setup

Eufy - Encryption

Selective execution

- Debugger (gdb, radare2) ✗
  - Allows for targeted execution ✓
    - With concrete inputs
  - Doesn't work on cross-architecture binaries ✗
  - Cross-compiled debugger to use on device ✓

Eufy - Encryption

Selective execution

- Emulation ✗
  - Execute cross-architecture binaries ✓
  - No selective execution ✗
  - Complex ✗
    - NVRAM calls
    - Peripherals and sensors

Eufy - Encryption

Selective execution

- Binary lifting ✗
  - Makes it architecture-agnostic ✓
    - Execute lifted binary
  - No selective execution ✗

Eufy - Encryption

Selective execution

- dAngr
  - Debugger for angr
    - Combines **binary lifting** and **debugging**
      - VEX IR
      - Debugger
  - ~=Cross-architecture symbolic GDB
  - Simplifies the use of angr
    - CLI

Eufy - Encryption

Selective platform-independent execution

```c
#include <stdio.h>
#include <string.h>

int processMessage(const char* in, int cnt, char* out){
   if (cnt > 0) {printf("count: %d", cnt);}
   int in_len = strlen(in);
   for(int i = 0; i < cnt; ++i){
      for (int j = 0; j< in_len; ++j){
         out[j+i*in_len] = in[j];
      }
   }
   out[cnt * in_len] = '\0'; // Null-terminate the output array after all characters are copied
   printf("p1: %s\n", out);
   return cnt*in_len;
}

int main() {
   char out[2 * 3 + 1]; // Adjusted size to accommodate the result and null terminator
   int i = processMessage("abc", 2, out); // Adjusted cnt parameter to 2
   printf("p2: %s\n", out);
   return 0;
}
```

Eufy - Encryption

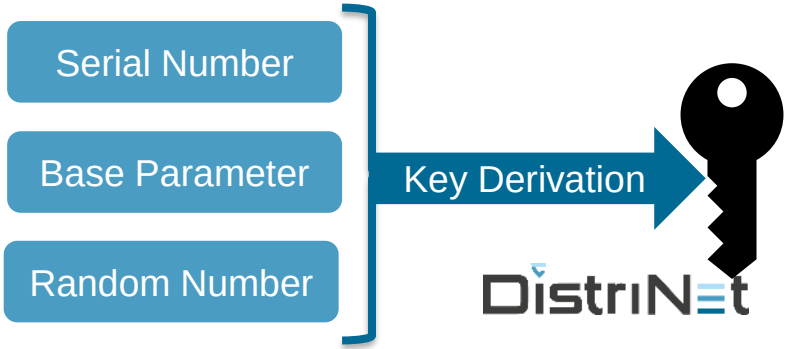
Selective platform-independent execution

- dAngr
  - Relay the start of execution to a selected address
  - Setting and retrieving registers or memory
  - Debug over logic blocks
  - Stops at
    - Breakpoints
    - Branch points

Eufy - Encryption

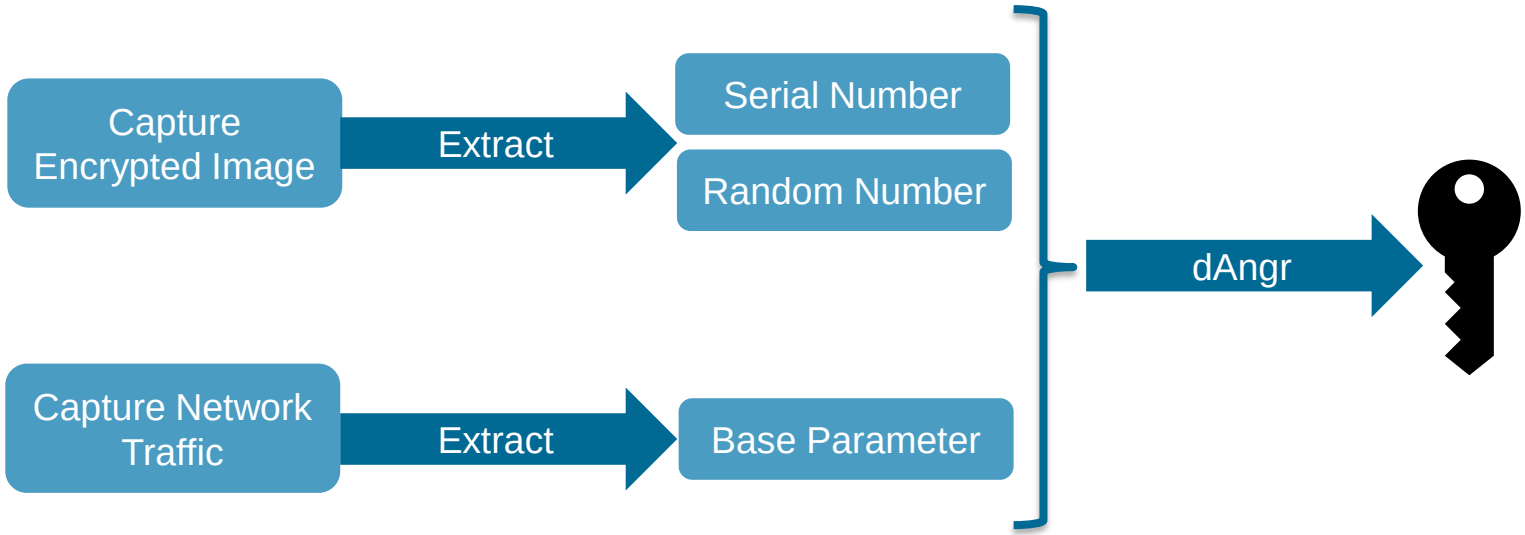
Selective platform-independent execution

- Leverage dAngr to replicate the AES key
  - Call the encryption function
    - Serial and base parameter as parameters
    - Hook the random number
  - Get AES key as return value



Eufy - Encryption

Selective platform-independent execution



DistriNet

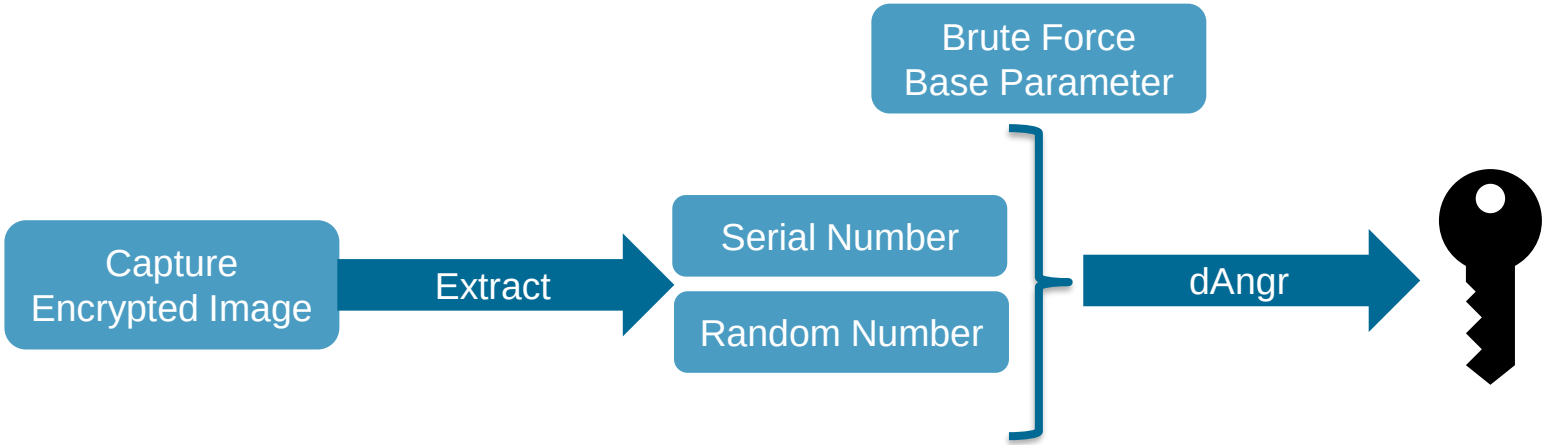
Eufy - Encryption

- Analysis of the encryption method
  - Only small part of the “Base Parameter” used
  - Brute forceable

Algorithm 2 `getPPCSSuffix`(*PPCS_ID*)

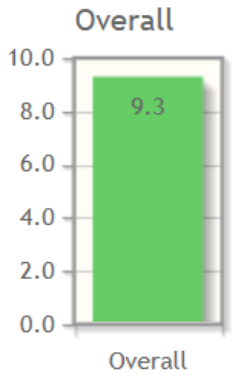
1: $s = PPCS\_ID[0:15].split('-')[1]$
2: $sfx = \texttt{int}(s[0]) + \texttt{int}(s[1]) + \texttt{int}(s[3]) + \texttt{int}(s[5])$
3: **if** $sfx < 5$ **then**
4: $\quad sfx = sfx * 2$
5: **end if**
6: **return** sfx

Eufy - Encryption



Disclosure

- Disclosed June 2023
- Patched Juli 2024
- CVE-2023-37822



Back

Version History

4.8.70 2d ago
1. Upgraded advanced encryption protocol for enhanced security
2. Fixed some bugs

4.8.60 3w ago
1. Added new toggle switch "Quick Focus Tap" to SoloCam S340/Floodlight Cam E340/IndoorCam S350 more

4.8.50 1mo ago
1. Enhanced pan-tilt experiences for 4G&Wi-Fi Spotlight Cam
2. Fixed some bugs

4.8.42 1mo ago
1. Fixed some bugs

4.8.41 1mo ago
1. Fixed some bugs

4.8.40 1mo ago
1. Added 4G Starlight Camera's Pan&Tilt calibration, and opitimized the Wi-Fi connection experience. more

Messages

Top System

Update 16:50
Device HB2 TV has been updated successfully.

Firmware 3.3.4.1h
1. Enhance key generation, data encryption, and encryption protocol for stronger security.
2. Fixed some bugs.

Update Yesterday 04:02
Device HB2 on desk has been updated successfully.

firmware 3.3.3.7h:
1. Disable UART access prior to booting through an update to U-Boot.
2. Enhance Homebase's WPA2 password generation algorithm.
3. Limit network access to Homebase Wi-Fi exclusively when the unauthorized device is connected to Homebase.

Update Yesterday 02:41
Device HB2 TV has been updated

DistriNet

Insights

- DIY Solutions
- Unique passwords
  - Compliance
- Security vs usability
  - Vulnerability or feature
- Solution
  - Industry-standard, vetted protocols
  - Reference architectures
  - Proof of concepts

Conclusion

- Undermine the security of the complete ecosystem
  - Combine distinct attack vectors
- Introduced dAngr
  - Augments manual reverse engineering

Read the Paper!

DistriNet

Thank you!

Use dAngr!

It’s open source!

(And useful)