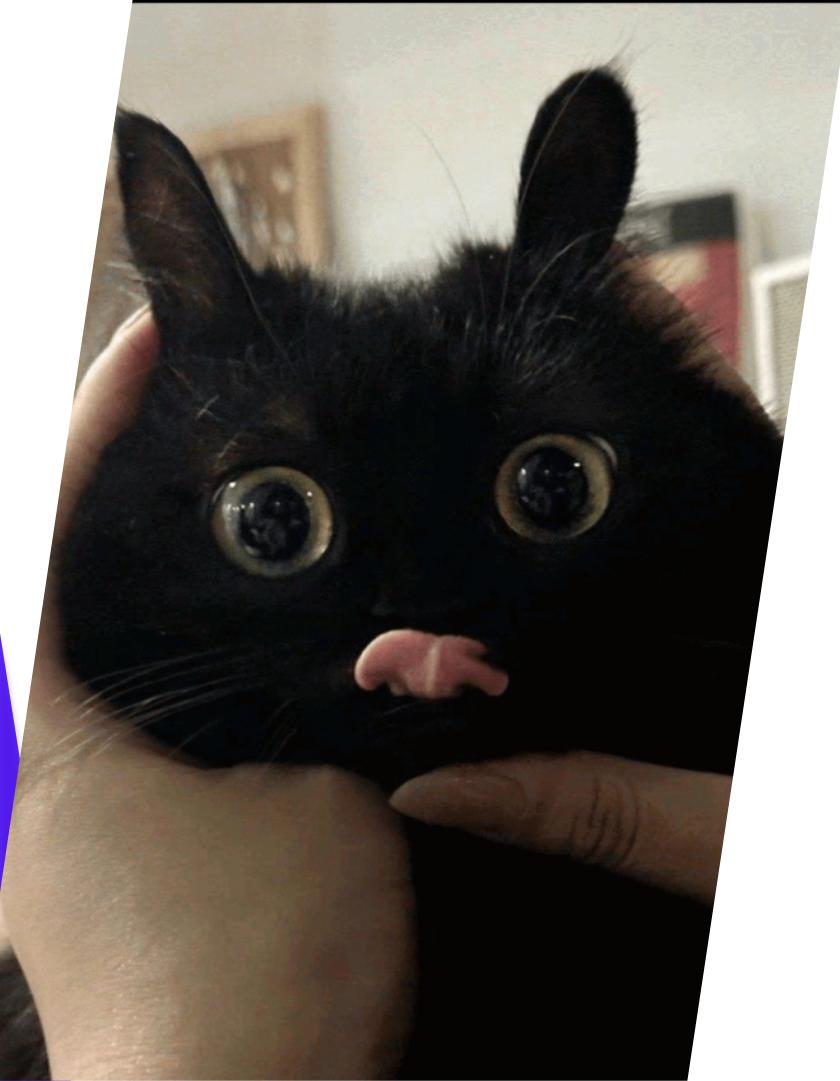


OzSec 2025

Isaiah Davis-Stober
Super cool guy





What I'm going to cover

- ▶ What is firmware?
- ▶ The IoT epidemic
- ▶ How to acquire firmware samples
- ▶ Dynamic and static analysis

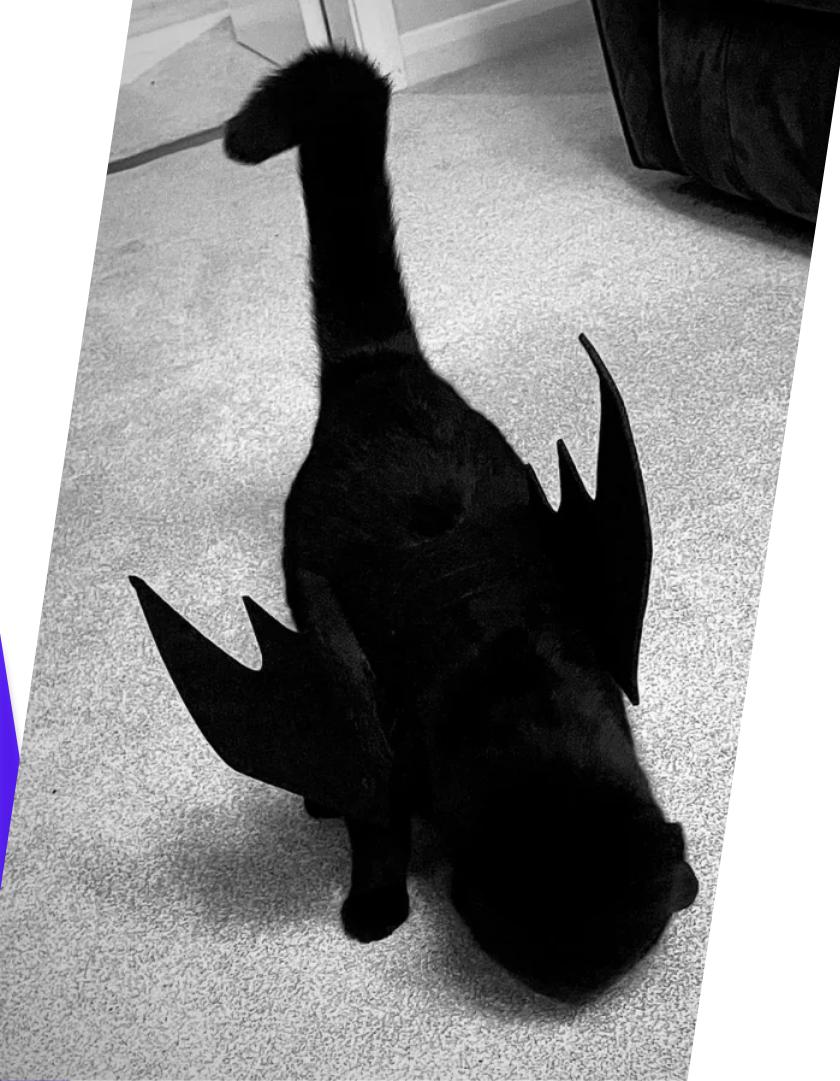
Who am I?

Experience:

I've been tearing apart (and sometimes putting back together) electronics since I was born, and tearing apart IoT devices for the last couple of years

Things I like:

- ▶ Misusing computers
- ▶ Cats
- ▶ Lock picking/bypassing
- ▶ Tearing apart whitelabel IoT devices



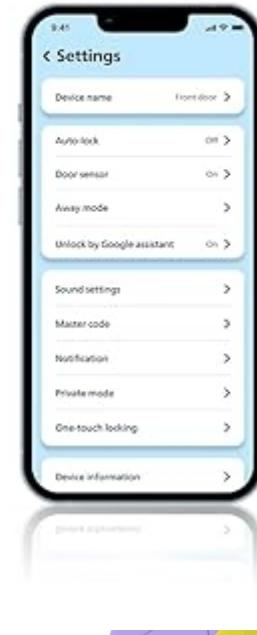
What is “IoT”?

It's the Internet of Things
babyyyyy

OzSec 2025



IoT Examples



OzSec 2025

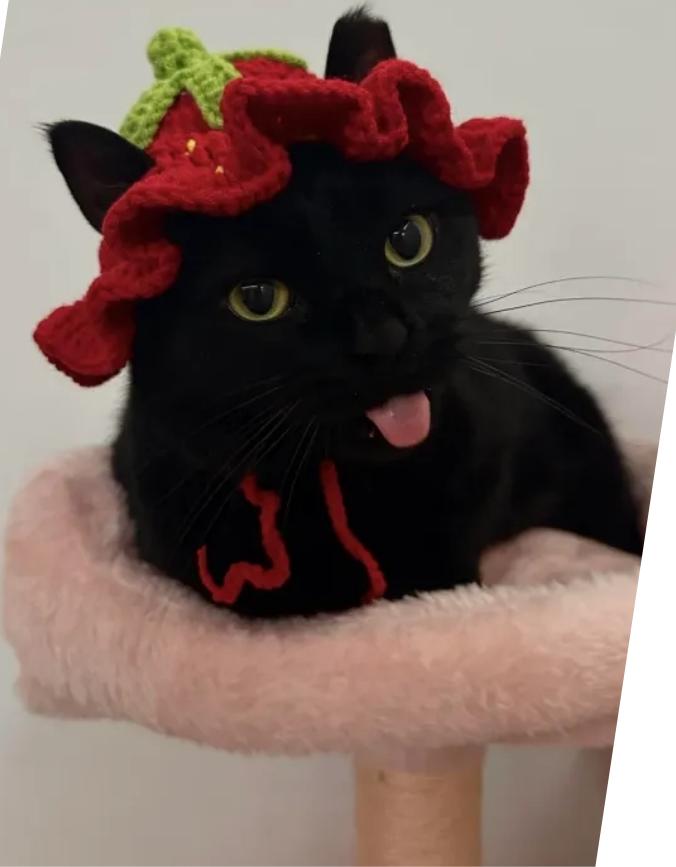


What is Firmware?

- ▶ Lowest level of software
- ▶ Interacts directly with the hardware

Familiar operating systems

Key differences between embedded Linux and other operating systems you are probably familiar with

A black cat with yellow eyes is sitting on a light-colored, fluffy surface. It is wearing a red and green knitted hat that looks like a small Christmas tree or a festive wreath. The cat has its tongue slightly out. The background is plain and light.

Why embedded is so different

- ▶ Different chip architecture
- ▶ Less compute power
- ▶ Super low level
- ▶ Interacting directly with hardware

Why firmware so interesting

- ▶ Lowest level means if you control it you are god
- ▶ It is often neglected/ignored



Why is it a fun target?

- ▶ Low level is fun
- ▶ I don't have to fight an AV
- ▶ No one configures things correctly

IoT epidemic and why it's stupid

- ▶ Every product is the minimum viable product
- ▶ Toasters were not meant to have an internet connection

i square to god



Quick rant
about
“smart”
devices



Privacy concerns

OzSec 2025



I just want it
to work, why
does it need
an account?

A black cat is sitting on a light-colored surface, wearing a green knitted hat. The hat has two small ears on top and a yellow string hanging down its front. The cat's eyes are wide and looking towards the camera.

Firmware acquisition

- ▶ Download
- ▶ Extraction

Downloading firmware

- ▶ Manufacture website
- ▶ Open S3 buckets
- ▶ Some very nice person who is distributing a copy in a legally questionable way
- ▶ FTP servers
- ▶ OTA update servers that only do useragent checks

http://fw.ajcloud.net/01.10711/gQdotgSHpnXq4JXEap4Nuw_ota_firmware_01.10711.11.01.pkg

For example: <ftp://ftp2.dlink.com/PRODUCTS/DIR-882/REVA/>



Bootloader

OzSec 2025



Common bootloaders

- ▶ U-Boot
- ▶ Barebox
- ▶ RedBoot
- ▶ RT-Thread



Extracting firmware

- ▶ UART
- ▶ JTAG
- ▶ Flash extraction (chip on and chip off)

UART (Universal Asynchronous Receiver/Transmitter)

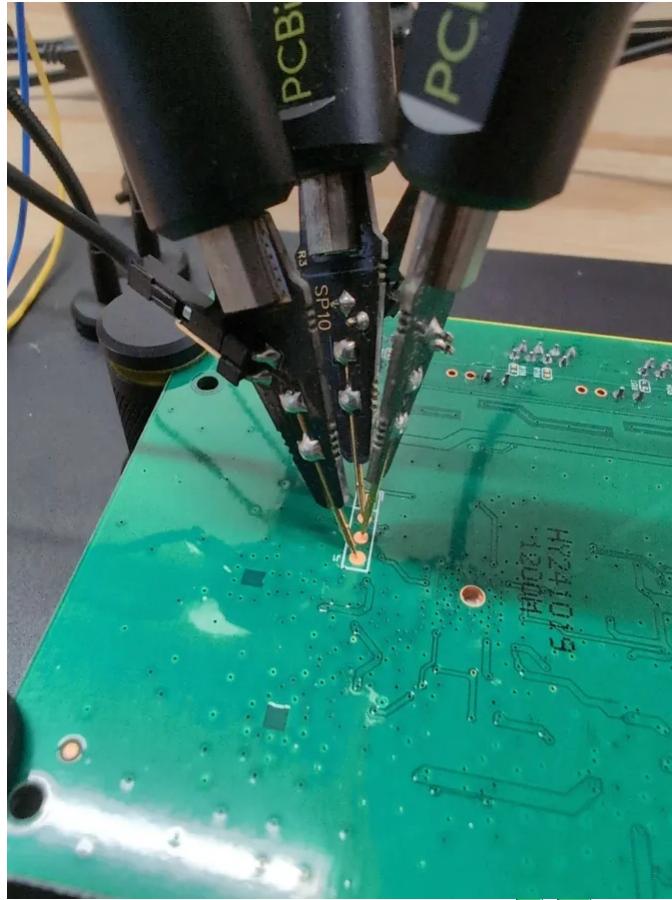
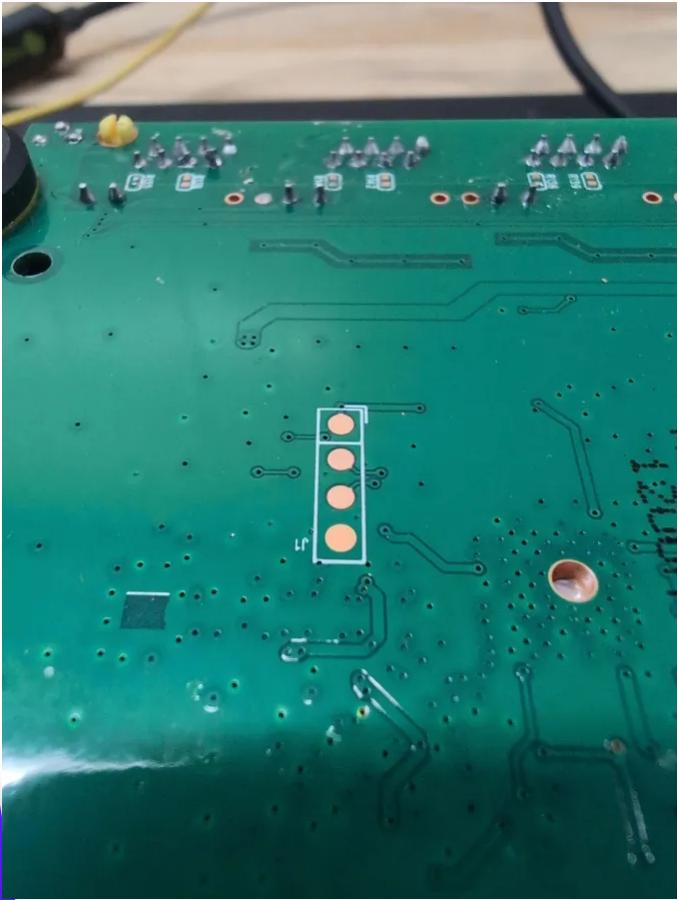
RX -> TX

TX -> RX

GND -> GND

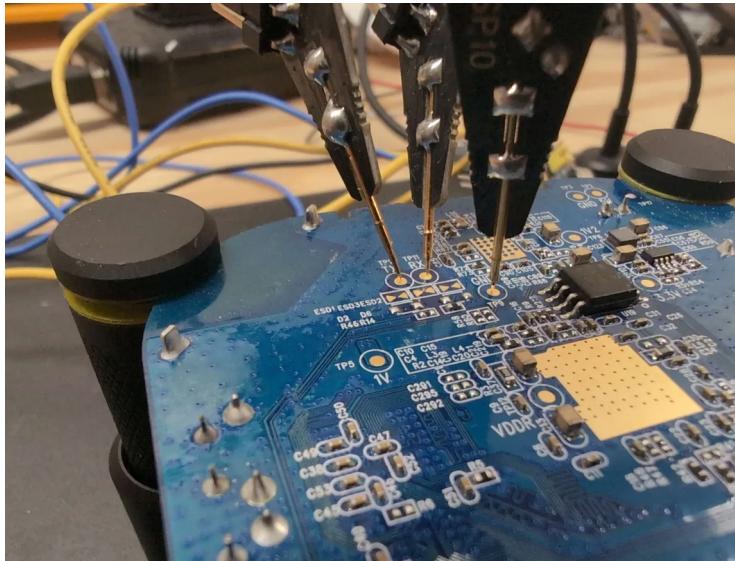
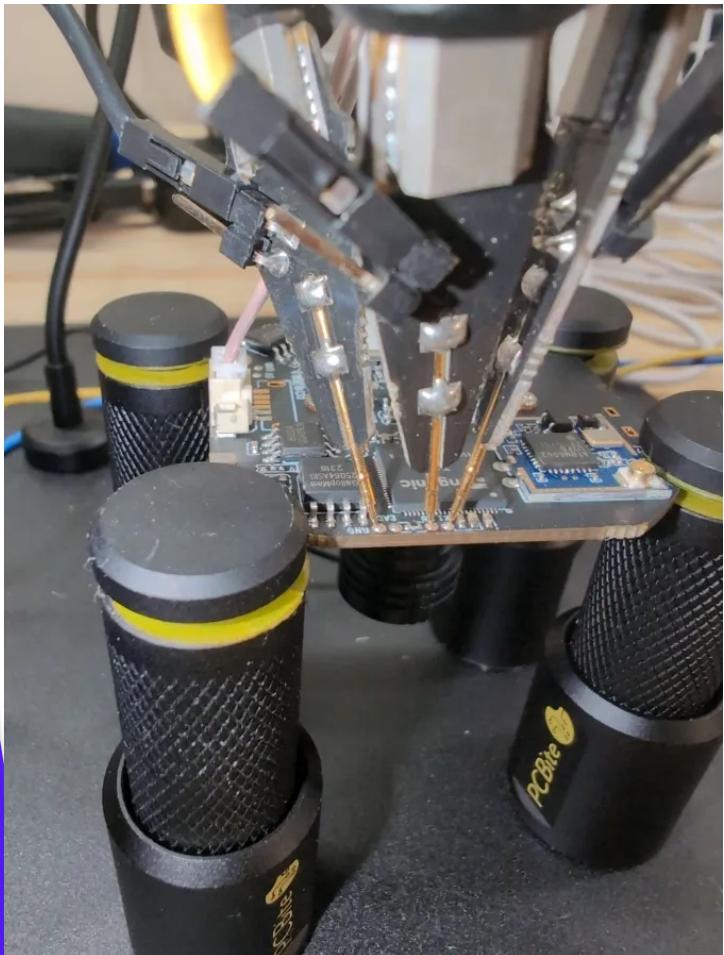
Common baud rates:

- 115200
- 57600
- 38400
- 9600



OzSec 2025





OzSec 2025



JTAG (Joint Test Action Group)

- ▶ Debugging interface similar to UART
- ▶ More pins
- ▶ More in depth
- ▶ Not just a serial terminal

```
BusyBox v1.27.2 () built-in shell (ash)

-----run profile file-----

[0000000779][W] IOT TEST:my_really_cool_wifi_password, IOT TEST:my_really_cool_wifi_password

[0000000802] firmware version: 1.7.3
[0000000802] not update firmware
[0000000813] fua_video_capture_jpg start
mount: mounting none on /sys/kernel/debug failed: No such file or directory
root@(none):/# [0000000823] get_wifi_data waitting..., register_flag 0
[0000000831][W] WAKEUP_TYPE_POWERON

[0000000882] fua_video_capture_jpg to /tmp/snap.jpg
[0000000897][I] common_i_pcm8kto16k /tmp/sound8k.pcm /tmp/sound.pcm OK, len 28800
[0000000897] play_audio_amr_2 /tmp/sound.pcm 100.
[0000000942] fua_set_day_night_mod day_night_mod 0
[0000000942][I] device_test_video_routine
[0000000942][I] device_test_routine wait usb connect
[0000000998] fua_audio_play_set_volume 100
[0000001042] change to day mode
? ? ? ? " " ? @

? ? ? @
    [54]readcmd 11
[56]SUNXI_SPI_DEFAULT_CLK = 7000000[54]readcmd 11
[56]SUNXI_SPI_DEFAULT_CLK = 7000000[377]#
ciapp start...
mount jffs2
mount jffs2 over
[000000492][W] ciapp(v:241) run build_2023/4/14:Apr 1 2024 06:00:08
```

Checking boot log for information

OzSec 2025



```
--RealTek(RTL8196E)at 2015.01.06-18:13-0800 v1.6 [16bit](400MHz)
bootbank is 1, bankmark FFFFFFF0
check_image_header  return_addr:05010000 bank_offset:00000000
no sys signature at 00010000!
no sys signature at 00020000!
no rootfs signature at 000E0000!
no rootfs signature at 000F0000!
Jump to image start=0x80500000...
return_addr = 05030000 ,boot bank=1, bank_mark=0xffffffff...
decompressing kernel:
Uncompressing Linux... done, booting the kernel.
done decompressing kernel.
start address: 0x80003480
CPU revision is: 0000cd01
Determined physical RAM map:
 memory: 02000000 @ 00000000 (usable)
Zone PFN ranges:
  Normal  0x00000000 -> 0x00002000
Movable zone start PFN for each node
early_node_map[1] active PFN ranges
  0: 0x00000000 -> 0x00002000
Built 1 zonelists in Zone order, mobility grouping on. Total pages: 8128
Kernel command line: console=ttyS0,38400 root=/dev/mtdblock1
icache: 16kB/16B, dcache: 8kB/16B, scache: 0kB/0B
NR_IRQS:48
PID hash table entries: 128 (order: 7, 512 bytes)
console handover: boot [early0] -> real [ttyS0]
Dentry cache hash table entries: 4096 (order: 2, 16384 bytes)
Inode-cache hash table entries: 2048 (order: 1, 8192 bytes)
Memory: 26420k/32768k available (2566k kernel code, 6348k reserved, 459k data, 108k init, 0k highmem)
Calibrating delay loop... 398.95 BogoMIPS (lpj=1994752)
Mount-cache hash table entries: 512
net_namespace: 524 bytes
NET: Registered protocol family 16
bio: create slab <bio-0> at 0
```

Flash dumping

- ▶ Types of flash
- ▶ Tools
- ▶ Methods

Types of flash interfaces

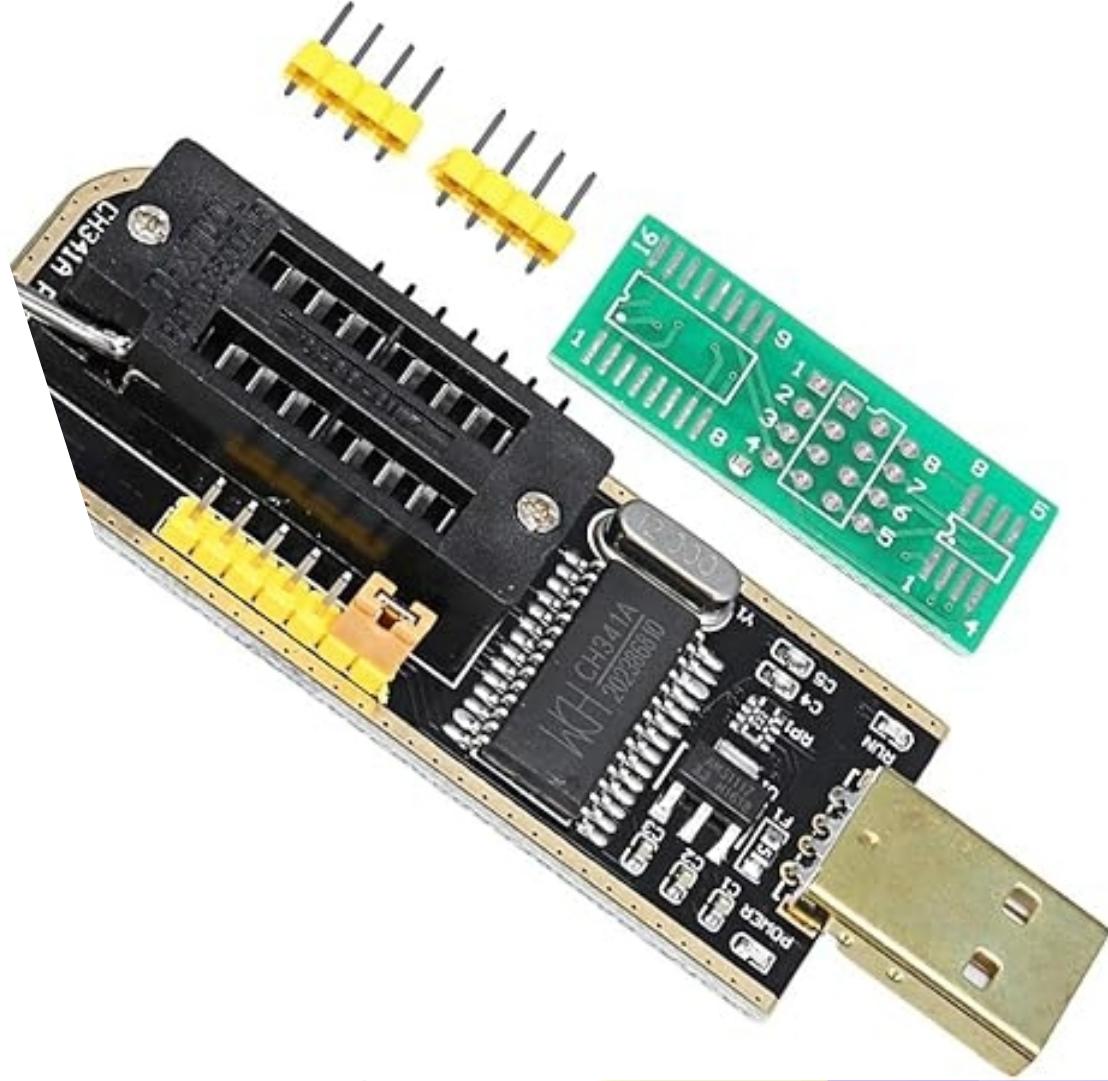
- ▶ SPI (Serial Peripheral Interface)
- ▶ QSPI (Quad Serial Peripheral Interface)
- ▶ I2C (Inter-Integrated Circuit)

Tools for dumping flash

- ▶ Hardware
 - Xgecu
 - CH341a
- ▶ Software
 - Flashrom
 - minipro

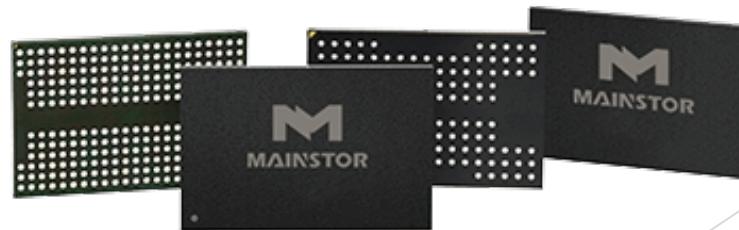
CH341a

- ▶ Easiest tool for using the CH341a programmer which is \$14 on Amazon
- ▶ Works on just about every chip I've tried*
- ▶ *dumping, it cannot always write them



Methods

- ▶ Chip on
 - Easy
 - No Soldering
 - Difficult with larger chips
 - Can only do packages with exposed pins
- ▶ Chip off
 - Some soldering
 - Extremely consistent
 - Can do much larger chips
 - Can do BGA packages

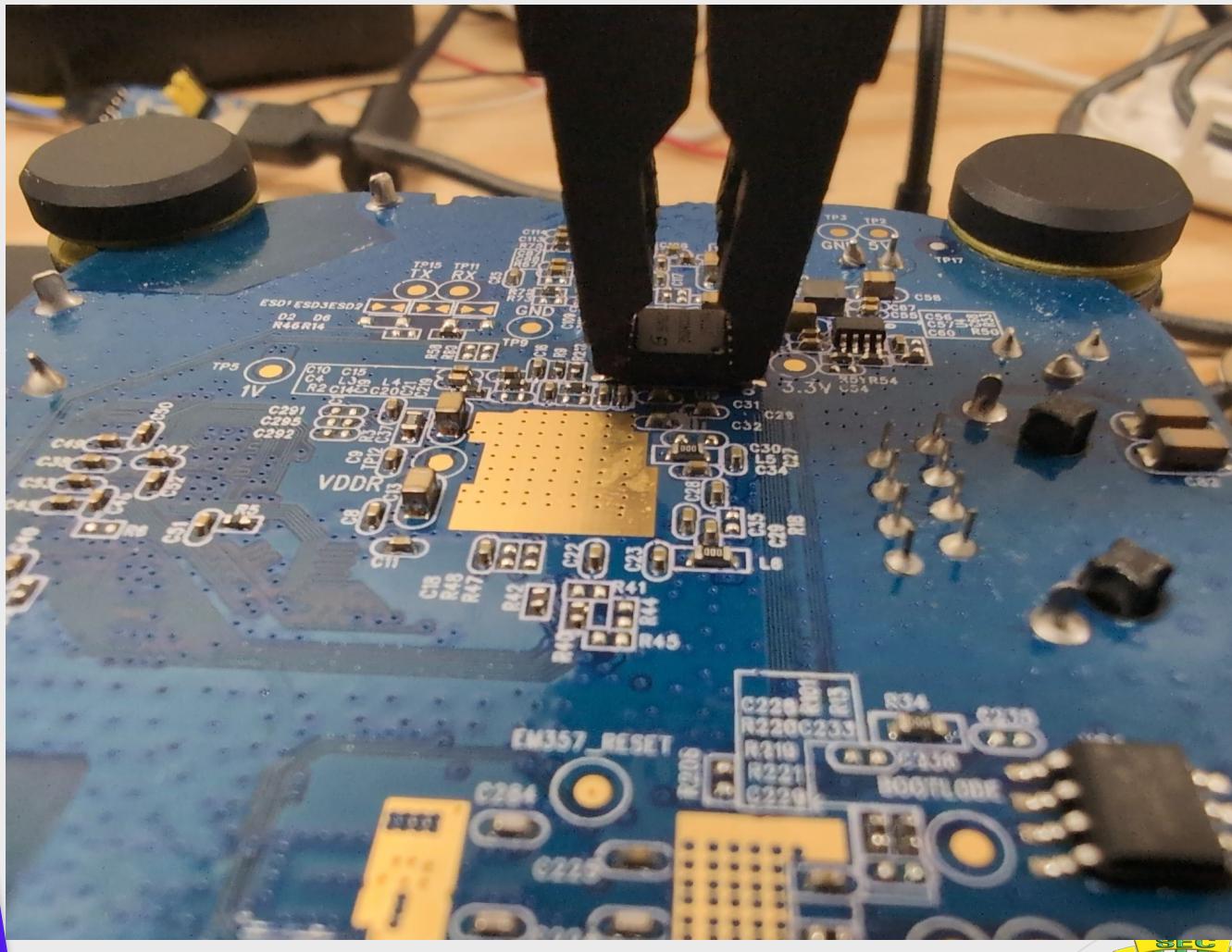


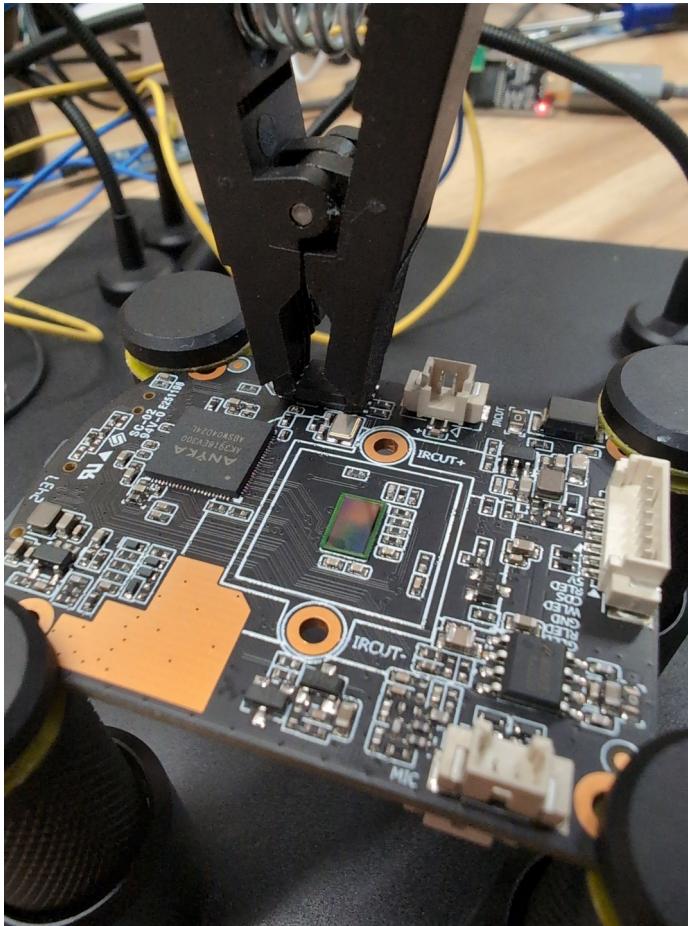
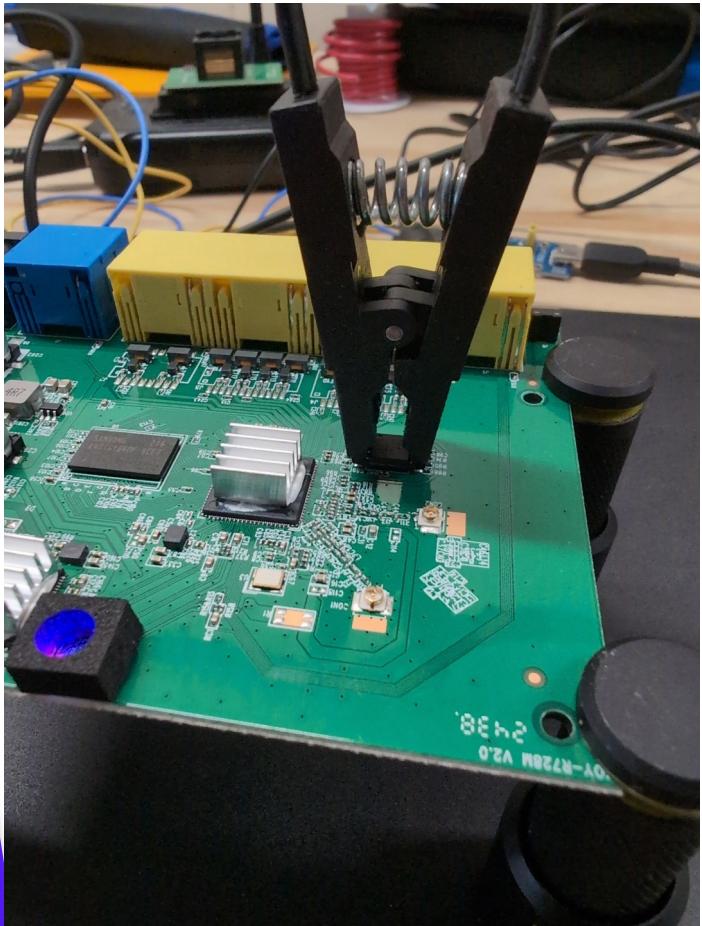
(This is a BGA package)

Chip on flash dumping

Dumping the firmware straight off of the flash module, without taking it off the board

- ▶ SOP8 clip
- ▶ CH341a and flashrom





OzSec 2025



Chip off flash dumping

- ▶ Adapter for whatever package you are dumping
- ▶ Minor soldering skills



OzSec 2025





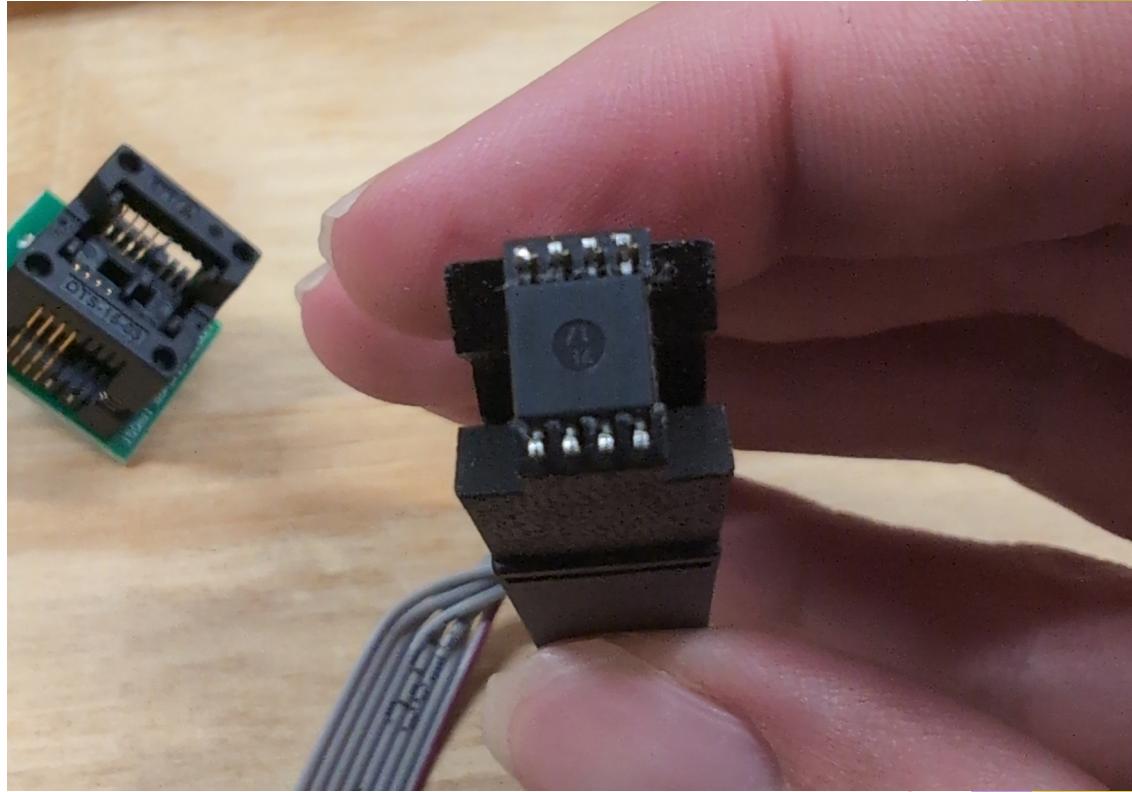
OzSec 2025

OZ
SEC



OzSec 2025





OzSec 2025



Using Flashrom

- ▶ Amazing for SPI flash

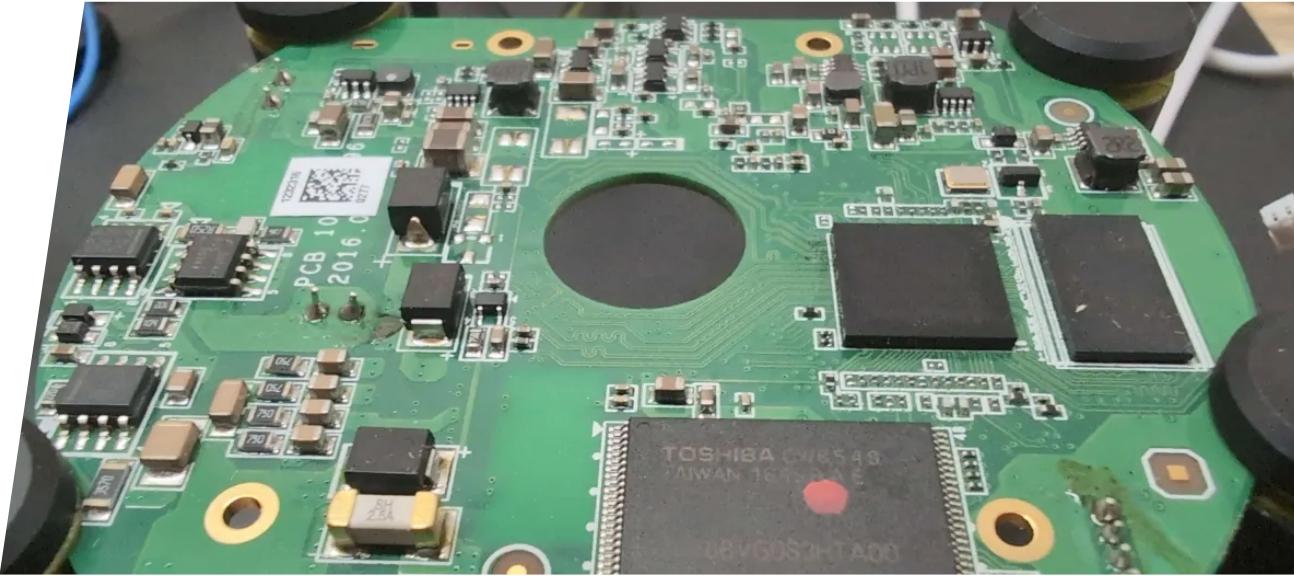


XGecu

- ▶ Amazing for everything else

OzSec 2025





Main use

OzSec 2025





Analyzing firmware

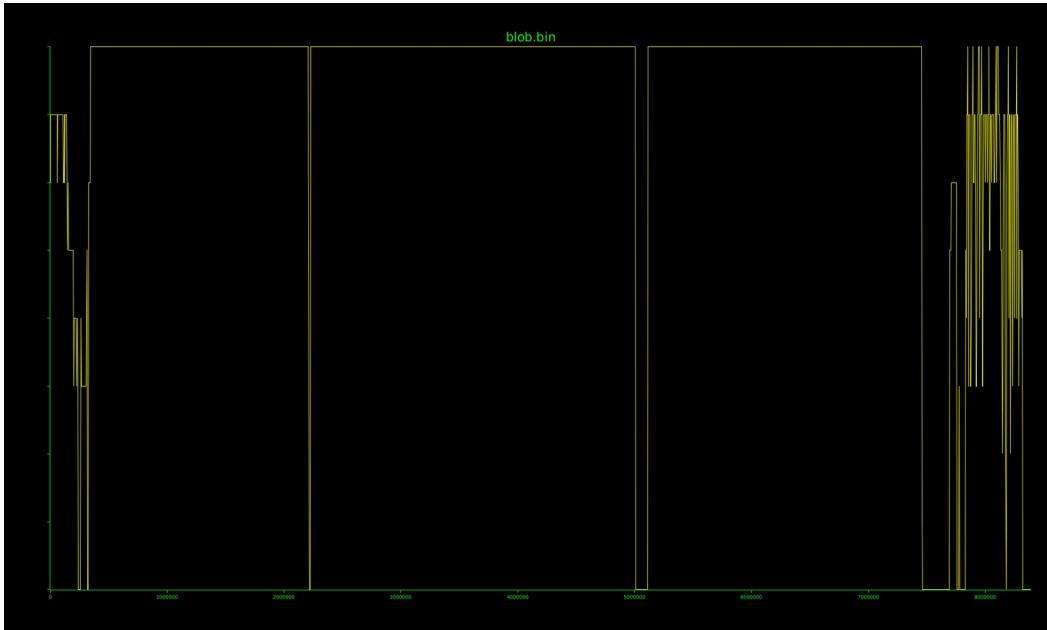
- ▶ Static vs dynamic analysis
- ▶ Common embedded systems filesystems

Checking if the dump is encrypted

- ▶ Check entropy with binwalk
- ▶ Checking what different parts of the file look like with binvis.io

Binwalk -E

Probably not encrypted

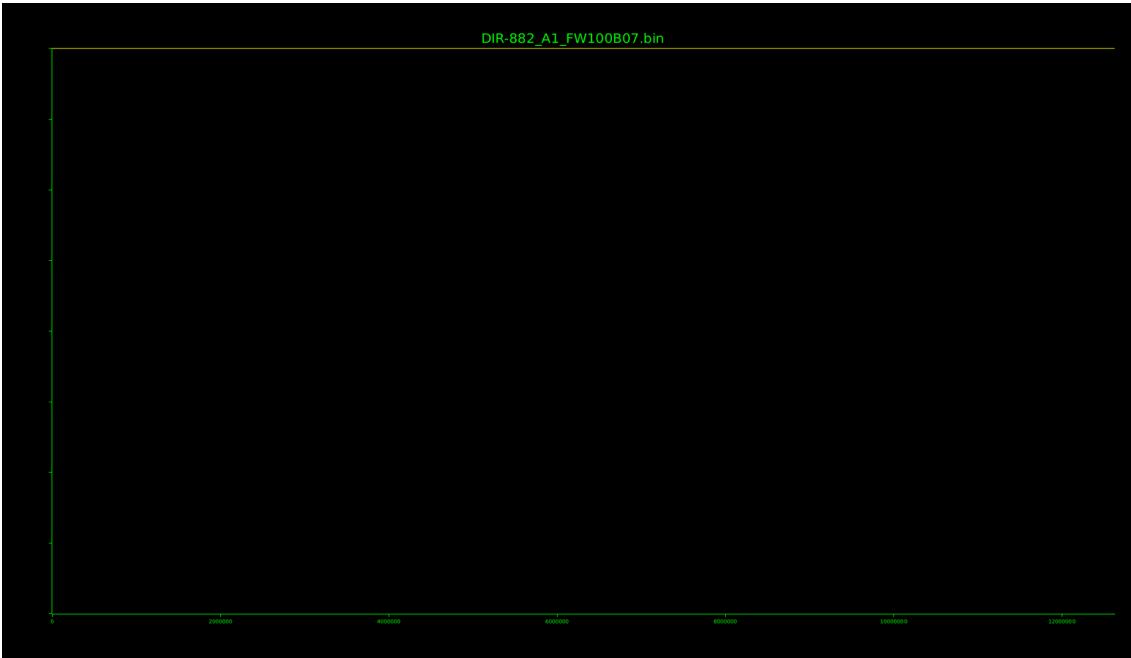


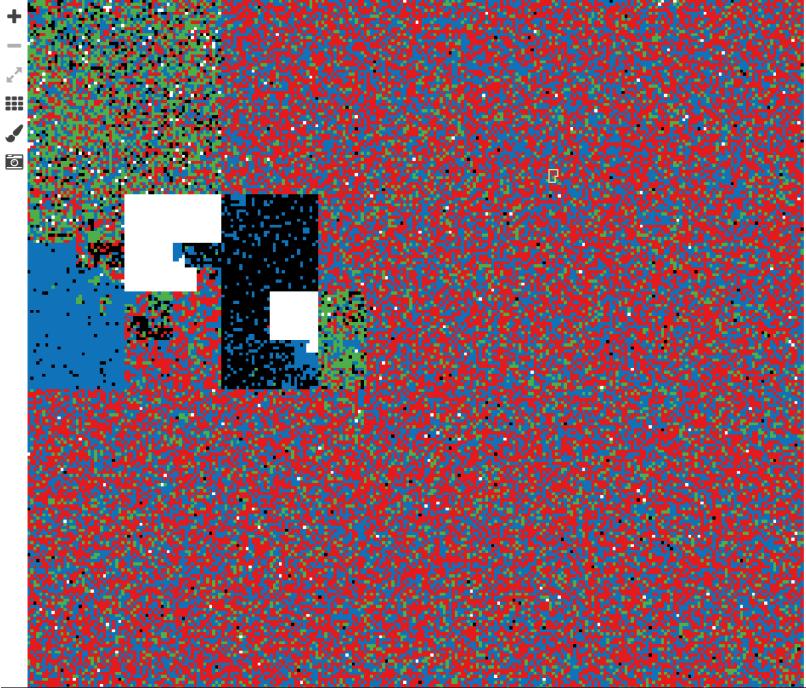
OzSec 2025



Binwalk -E

Almost certainly encrypted





hex	dec	blob.dump ▾
0x093680	58	7b 25 12 cb 27 dd b4 a0 fc 5d 58 0e 8b f0 41 X{%. '... . . .}X...A
0x093680	6d	49 a1 7b fe 75 8c f1 12 c5 35 6b 18 6d 52 285 .0.(_.u. . . .5..mr.
0x093680	24	16 1e 0d b4 ab c7 3d 89 95 6b e6 9c d7 e9 45 \$...-.-. . . .A
0x093680	59	20 0f 0d 09 0d m.4.P... .2...3..
0x093680	71	28 8e 86 f3 67 77 c9 88 32 01 01 01 01 01 01 01 01 q{...g. 4...n..
0x093680	47	2d 12 6d 90 2d fd e2Mc. 0.m..
0x093680	2e	9d 80 08 e9 44 63 f3 77 3e ee 6a b1 6a dd 2fM4. v>..j.j./
0x093680	01	3a e9 ab e7 02 b5 25 c2 24 6e 36 5d 88 77 62 \$nol.wb
0x093680	d4	ae 6a e5 7f 4e 34 cc 77 3e ee 6a b1 6a dd 2fM4. v>..j.j./
0x093680	c4	ea c9 d4 f9 fd 0c 6c b7 f4 80 bb 40 38 c6 d9l88..
0x093710	e3	0b 69 01 cb 5d c0 38 1c f8 83 1f dd 6f 02 45 o.E
0x093720	7c	9b 34 53 9a 09 49 3f 91 1e a8 13 23 22 eb 41 .45..I7 #.A
0x093730	14	55 83 5f e8 31 28 26 3e a4 45 3b 8f e5 59 cd ..U.._!G >E..Y..
0x093740	92	99 29 61 63 62 e9 1d 8d 7b 27 fe 62 e4 c3 ..)acb.. ..('..b..
0x093750	e9	e3 05 ef d5 00 d1 76 e7 7b 78 5b 27 47 06 f2v ..{x}'G..
0x093760	fe	b9 04 12 a2 ba a9 50 f4 97 23 29 c3 00 38 21P ..#)..8!
0x093770	d7	ca b7 75 ea 25 e6 26 d8 46 20 41 1b 7b 51 99 .. .u.%& ..F ..@..
0x093770	54	10 12 9a 67 28 cf 2f b6 2a 1f 18 04 8e 97 51 T...g(. / .. * .. .0
0x093770	57	70 23 23 39 bf 61 70 07 8f 20 41 ac f9 b9 p##9.ap .. . A...
0x093780	52	f2 a7 cb fa f0 f2 c0 91 bd 15 17 ac 8a c4 ac R.....
0x093780	0a	69 ca 8e 12 57 f4 08 28 70 61 fb 8c 82 37 4c .i..W.. ..(pa...7L

byteclass range
0x00 [0 - 8388608] export
low 8mb / 8mb
ascii
high
0xffffffff

binvis.io

OzSec 2025



Static analysis

- ▶ Binwalk
- ▶ Strings
- ▶ Ghidra
- ▶ xxd

- ▶ FACT (Firmware Analysis and Comparison Toolkit)

Determining the type of system you're working with

- ▶ CPU architecture
 - MIPS
 - ARM
- ▶ What the firmware might be based on
 - Linux
 - FreeRTOS
 - Something else

Extracting the filesystems and directory structure

```
[+09:50:54 on main ✘ ↵]→ binwalk -e blob.bin →(Fri,Apr25)←  
/home/neko/firmware/sengled/extractions/blob.bin  
  
DECIMAL          HEXADECIMAL      DESCRIPTION  
-----  
4896             0x1320          LZMA compressed data, properties: 0x5D,  
                                dictionary size: 8388608 bytes, compressed  
                                size: 18576 bytes, uncompressed size: 60912  
bytes  
65552            0x10010         bzip2 compressed data, total size: 129090 bytes  
206872           0x32818         LZMA compressed data, properties: 0x5D,  
                                dictionary size: 8388608 bytes, compressed  
                                size: 920470 bytes, uncompressed size: 3207476  
bytes  
1245184          0x130000        SquashFS file system, little endian, version:  
                                4.0, compression: lzma, inode count: 518, block  
                                size: 131072, image size: 1770878 bytes,  
                                created: 2038-01-29 01:48:48  
4259856          0x410010        bzip2 compressed data, total size: 129090 bytes  
4401176           0x432818        LZMA compressed data, properties: 0x5D,  
                                dictionary size: 8388608 bytes, compressed  
                                size: 920470 bytes, uncompressed size: 3207476  
bytes  
5439488          0x530000        SquashFS file system, little endian, version:  
                                4.0, compression: lzma, inode count: 518, block  
                                size: 131072, image size: 1770878 bytes,  
                                created: 2038-01-29 01:48:48  
  
[+] Extraction of lzma data at offset 0x1320 completed successfully  
[+] Extraction of bzip2 data at offset 0x10010 completed successfully  
[+] Extraction of lzma data at offset 0x32818 completed successfully  
[+] Extraction of squashfs data at offset 0x130000 completed successfully  
[+] Extraction of bzip2 data at offset 0x410010 completed successfully  
[+] Extraction of lzma data at offset 0x432818 completed successfully  
[+] Extraction of squashfs data at offset 0x530000 completed successfully  
  
Analyzed 1 file for 101 file signatures (226 magic patterns) in 449.0 milliseconds
```

OzSec 2025



Common File systems

Read only

- ▶ SquashFS

Writeable

- ▶ JFFS2 (Journaling Flash File System v2)

Combining them

- ▶ OverlayFS

Extracting those common filesystems

SquashFS

- ▶ UnsquashFS
- ▶ Sasquatch (patch for unsquashfs)

JFFS2

- ▶ Jefferson

Directory layout

- ▶ exa -T
- ▶ ls -R

```
└── udev
    ├── rules.d
    └── udev.conf
├── files
└── pseudo_init
├── home
└── lib
    ├── firmware
    ├── ld-musl-armhf.so.1 -> libc.so
    ├── libc.so
    ├── libgcc_s.so.1
    ├── libstdc++.so.0 -> libstdc++.so.6.0.22
    ├── libstdc++.so.6.0.22
    └── libstdc++.so.6.0.22-gdb.py
    └── modules
        ├── grace.ko
        ├── libcomposite.ko
        ├── lockd.ko
        ├── nfs.ko
        ├── nfsv2.ko
        ├── nfsv3.ko
        ├── sunrpc.ko
        ├── sunxi_hci.ko
        ├── usb_f_fs.ko
        └── usb_stream.ko
        └── usbcore.ko
└── mnt
    ├── app
    ├── extsd
    ├── exUDISK
    ├── SDCARD
    └── sdcard
    └── UDISK
    └── overlay
    ├── proc
    ├── pseudo_init
    ├── rdinit -> pseudo_init
    ├── root
    ├── run_usb_adb
    └── sbin
        ├── getty -> ../bin/busybox
        ├── hwclock -> ../bin/busybox
        ├── ifconfig -> ../bin/busybox
        └── init -> ../bin/busybox
```

Determining how the filesystems are mounted

- ▶ fstab
- ▶ inittab
- ▶ Whatever other init script they are using

```
File: inittab.sh
15
16 # Startup the system
17 ::sysinit:/sbin/swapoff -a
18 # ::sysinit:/bin/mount -t tmpfs tmpfs /dev
19 ::sysinit:/bin/mkdir -p /dev/pts
20 ::sysinit:/bin/mkdir -p /dev/shm
21 ::sysinit:/bin/mount -a
22 ::sysinit:/bin/hostname -F /etc/hostname
23
24 # now run any rc scripts
25 ::sysinit:/etc/init.d/rcS
26
27 # Put a getty on the serial port
28 #console::respawn:-/bin/sh
29 console::respawn:/sbin/getty -L console 115200 vt100 # GENERIC_SERIAL
30
31 # Stuff to do for the 3-finger salute
32 ::ctrlaltdel:/sbin/reboot
33
34 # Stuff to do before rebooting
35 ::shutdown:/bin/umount -a -r
```



Locating files

- ▶ grep
- ▶ find
- ▶ rg (ripgrep)
- ▶ exa -T
- ▶ Guessing

Getting the BusyBox Version

```
strings -n 10 busybox | grep "BusyBox"
```

```
BusyBox v1.27.2 ()  
syslogd started: BusyBox v1.27.2
```

Looking for configuration files

- ▶ .conf
- ▶ .ini

```
└─(11:17:12 on main ✘ * * *)→ find . -type f -name "*.conf"
./etc/lld2d.conf
./etc/host.conf
./etc/ushare.conf
./etc/wscd.conf
./etc/boa/boa.conf
./etc/vsftpd.conf
./etc/samba/smb.conf
```

Finding misconfigurations in said conf files

- ▶ It helps to be somewhat familiar with the default configuration file for the service (if possible)
- ▶ Just read it
- ▶ Not every misconfiguration is actually exploitable or useful

Locating custom utilities and scripts

- ▶ A basic understand of Linux and common utilities goes a long way here in not wasting your time

First, BusyBox

- ▶ A useful utility for small embedded Linux machines
- ▶ Single utility
- ▶ Does all of the things
- ▶ Swiss Army Knife of Embedded Linux
- ▶ No one updates it

```
usr  
└── bin  
    [ -> ../../bin/busybox  
    [[ -> ../../bin/busybox  
    amp_shell  
    awk -> ../../bin/busybox  
    basename -> ../../bin/busybox  
    ciapp  
    ciconfig.ini  
    cksum -> ../../bin/busybox  
    config_network.sh  
    crontab -> ../../bin/busybox  
    cut -> ../../bin/busybox  
    dirname -> ../../bin/busybox  
    du -> ../../bin/busybox  
    env -> ../../bin/busybox  
    expr -> ../../bin/busybox  
    flock -> ../../bin/busybox  
    hexdump -> ../../bin/busybox
```

Just symlink

- ▶ It's that easy
- ▶ To find custom utilities just look for executables that aren't symlinks

```

File: pseudo_init

1 #!/bin/sh
2
3 MOUNT_ETC=1
4 MOUNT_OVERLAY=0
5
6 ##### functions #####
7
8 #mkfs.jffs2() <device in /dev/by-name>
9 mkfs_jffs2() {
10     [ -x /usr/sbin/mkfs.jffs2 ] \
11     && ! [ -x /sbin/mkfs.jffs2 ] \
12     && echo "Not Found /usr/sbin/mkfs.jffs2 or /sbin/mkfs.jffs2" \
13     && return 1
14
15     # format to jffs2
16     local erase_block=$(cat /proc/mtd \
17         | /bin/grep "$basename $1" \
18         | /usr/bin/awk '{print $3}')
19     /bin/mkdir -p /tmp/jffs2.dir/tmp
20     mkfs.jffs2 -p -e $erase_block -d /tmp/jffs2.dir \
21         -o /tmp/jffs2.img >/dev/null || return 1
22     /bin/dd if=/tmp/jffs2.img of=$1 || return 1
23     /bin/rm -rf /tmp/jffs2.img /tmp/jffs2.dir
24     return 0
25 }
26
27 mkfs.ubifs() {
28     mkfs.ubifs -x lzo -y "$1"
29 }
30
31 mount_etc() {
32     local etc_update=0
33     # if enable ota, do update
34     [ -f /etc/init.drc.ota-upgrade ] \
35     && source /etc/init.d/ota-upgrade
36
37 #local root_dev=$(readlink /dev/by-name/rootfs)"
38 local root_dev=$(readlink /dev/by-name/rootfs_data)"
39
40 # if mount failed, format.
41 case ${root_dev} in
42     /dev/mtdblock*)
43         # /bin/echo "mount jffs2"
44         /bin/mount -t jffs2 -o rw,sync /dev/by-name/rootfs_data /etc
45         # /bin/echo "mount jffs2 over"
46
47         [ -e /etc/etc_complete -a ! -e /etc/etc_need_update ] \
48             && return
49         # /etc/etc_complete and /etc/etc_need_update both exist, that means we just need to update
50         [ -e /etc/etc_complete -a -e /etc/etc_need_update ] && /bin/echo "do etc update" && etc_update=1
51
52         cp -arf /etc/ciconfig.inl /tmp/
53         cp -arf /etc/config_data.dat /tmp/
54         cp -arf /etc/other_data.dat /tmp/
55         /bin/umount /etc
56 }

```

Checking out custom scripts

- ▶ Usually as simple as reading them

Common things you might find in custom scripts

- ▶ Embedded credentials
- ▶ Possible endpoints
- ▶ OTA (Over The Air) update process
- ▶ telnetd initialization
- ▶ DropBear initialization
- ▶ Poor error handling
- ▶ Understanding of how the filesystems are setup/used
- ▶ True enlightenment

Finding hard coded credentials

- ▶ Passwd file
- ▶ Shadow file
- ▶ Keys
- ▶ Configuration files
- ▶ Custom application/script strings

```
jffs2-root
├── adb_profile
├── asound.conf
├── banner
├── banner.failsafe
├── ciconfig.ini
├── config_data.dat
├── crontabs
├── device_info
├── etc_complete
├── fstab
├── group
├── init.d
│   ├── adbd
│   ├── cron
│   ├── ntpd
│   └── rc.modules
│       └── rcS
├── inittab
├── mdev.conf
└── mtab -> /proc/mounts
openwrt_release
openwrt_version
other_data.dat
passwd
profile
rc.common
rc.d
└── K99adbd -> ../init.d/adbd
└── S50cron -> ../init.d/cron
└── S80adbd -> ../init.d/adbd
resolv.conf
shadow
shells
sysctl.conf
syslog.conf
tmp
udev
└── rules.d
└── udev.conf
```

Once you've found them

- ▶ Often hashed (but not always)
- ▶ Google the hash

File: shadow	
1	root:91rMizzGliXHM:1:0:99999:7:::
2	daemon:*:0:0:99999:7:::
3	ftp:*:0:0:99999:7:::
4	network:*:0:0:99999:7:::
5	nobody:*:0:0:99999:7:::

File: passwd	
1	root:\$1\$0WlvKUDR\$.yqcW5hBKvVJKCHQ4njdB/:0:0:root:/root:/bin/ash
2	daemon:*:1:1:daemon:/var:/bin/false
3	ftp:*:55:55:ftp:/home/ftp:/bin/false
4	network:*:101:101:network:/var:/bin/false
5	nobody:*:65534:65534:nobody:/var:/bin/false

For instance, Samba

	File: smbpasswd	
1	samba:500:E37D9D1B60CE6031F4EE5CAB3C10B45B:246D949F60611CFA928A476B3FF28B25:[U]:LCT-43D6003C:	

This one is “Shirley/”

Over The Air updates

A possible attack vector if there is no signing or security efforts put into the update process

```
31     mount_etc() {
32         local etc_update=0
33         # if enable ota, do update
34         [ -f /etc/init.d/rc.ota-upgrade ] \
35             && source /etc/init.d/ota-upgrade
36 }
```

telnetd

- ▶ It's telnet, what more do you want?
- ▶ Even if it is configured it might not actually be in started automatically

```
426    #/usr/sbin/telnetd &
427    /sbin/syslogd &
428    #hardcode but fast
429    #mount_etc_hardcode
430    #set_parts_by_name_hardcode
431    #mount_usr
432
433    exec /sbin/init
```

```
426    /usr/sbin/telnetd &
427    /sbin/syslogd &
428    #hardcode but fast
429    #mount_etc_hardcode
430    #set_parts_by_name_hardcode
431    #mount_usr
432
433    exec /sbin/init
```

SSH

- ▶ Often Dropbear
- ▶ Often super out of date

File: **dropbear**

```
1 config dropbear
2   option PasswordAuth 'on'
3   option RootPasswordAuth 'on'
4   option Port          '22'
5 #   option BannerFile  '/etc/banner'
```

Appalling script error handling

- ▶ It's always bash
- ▶ No one ever does proper error handling in bash

Weird and wacky filesystem setups

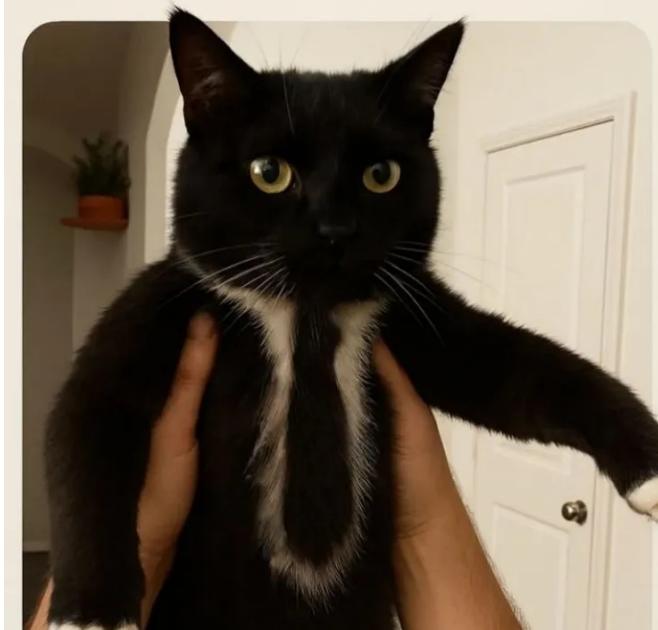
JFFS2 and SquashFS can be layered and can be used to do some funky stuff

Usually this is just OverlayFS, but sometimes something interesting will be done to handle OTA updates or factory resets

Checking out custom utilities

- ▶ Strings
- ▶ Ghidra (or IDA if you have money like that)
- ▶ xxd

sir put me down.
i am the manager



File: **base_conf.ini**

```
1 [auth]
2 key=1111111111111111
3 [USERINFO]
4 #压力测试使用 / 正常填写 Off 即可
5 PressureTest=Off
6 #新产测
7 NewfactoryTest=On
8 #定时重启
```

└─(11:24:11)→ bat ciconfig.ini

File: **ciconfig.ini**

```
1 [wlan]
2 keepalive_interval=15
3 dtim_interval=6
4 suspend_pm=2
5 ap_ssid=
6 ap_pwd=
7
8 [server]
9 web_server_url=https://api.v2.gdpx.com;
10 cmd_server_ip=120.24.87.105;
11 cmd_server_port=8888;
12 udp_server_ip=120.24.87.105;
13 udp_server_port=25050;
14 stun_server_ip=8.218.91.142;
15 stun_server_port=17051;
16 p2p_server_ip=47.242.63.121;
17 p2p_server_port=17051;
18 push_server_url=http://120.24.87.105:58720;
19 oss_from_id=9;
20 oss_endpoint=http://oss-cn-shenzhen.aliyuncs.com;
21 oss_bucket=sz-aiwit;
```

Common things to find in custom utilities

- ▶ Embedded credentials
- ▶ URLs/URIs
- ▶ IPs and ports
- ▶ API keys

OzSec 2025



F.A.C.T (Firmware Analysis and Comparison Toolkit)

- ▶ Great for basic static analysis of firmware samples
- ▶ You will still have to explore custom utilities and scripts yourself
- ▶ Really helpful to compare a new sample to see if you can expect something you've seen before

- ▶ https://fkie-cad.github.io/FACT_core/

AOQEE AOQEE_C1 - 1.0.0 (Security Camera)

Private Key Found critical CVE Linux Kernel 3.10.14

UID: c6232c06b09ec09a50b53cf971b094487b80a8bb5179f8d2b5a12814c1dc6ed_8388608

File Tree

blob.bin (3.00 MiB)

search file tree ...

Analysis Results

- [binwalk](#)
- [cpu architecture](#)
- [crypto material](#)
- [cve lookup](#)
- [cwe checker](#)
- [device tree](#)
- [exploit mitigations](#)
- [file hashes](#)
- [file type](#)
- [information leaks](#)
- [init systems](#)
- [interesting uris](#)
- [ip and url finder](#)
- [kernel config](#)
- [known vulnerabilities](#)
- [software components](#)
- [source code analysis](#)
- [unpacker](#)

Showing Analysis: cve lookup

Time of Analysis 2025-04-25 04:05:03

Plugin Version 0.1.0

Summary for Included Files

Item count	4
BusyBox 1.35.0	AOQEE AOQEE_C1 - 1.0.0 (Security Camera) /1D8000.squashfs /bin/busybox Size: 634.62 KiB , Type: application/x-executable 0
curl 7.65.3 (CRITICAL)	AOQEE AOQEE_C1 - 1.0.0 (Security Camera) /1D8000.squashfs /lib/libcurl.so.4 Size: 404.93 KiB , Type: application/x-sharedlib 0
hostapd 2.9 (CRITICAL)	AOQEE AOQEE_C1 - 1.0.0 (Security Camera) /1D8000.squashfs /bin/hostapd Size: 900.83 KiB , Type: application/x-executable 0
Linux Kernel 3.10.14 (CRITICAL)	AOQEE AOQEE_C1 - 1.0.0 (Security Camera) /40040.7z /2112d7d1850fc498184e65d03e31423544ae02fb550f2ccf1632be72d6a033a_8126400- Size: 4.36 MiB , Type: application/octet-stream 0

F.A.C.T.

OzSec 2025





Dynamic analysis

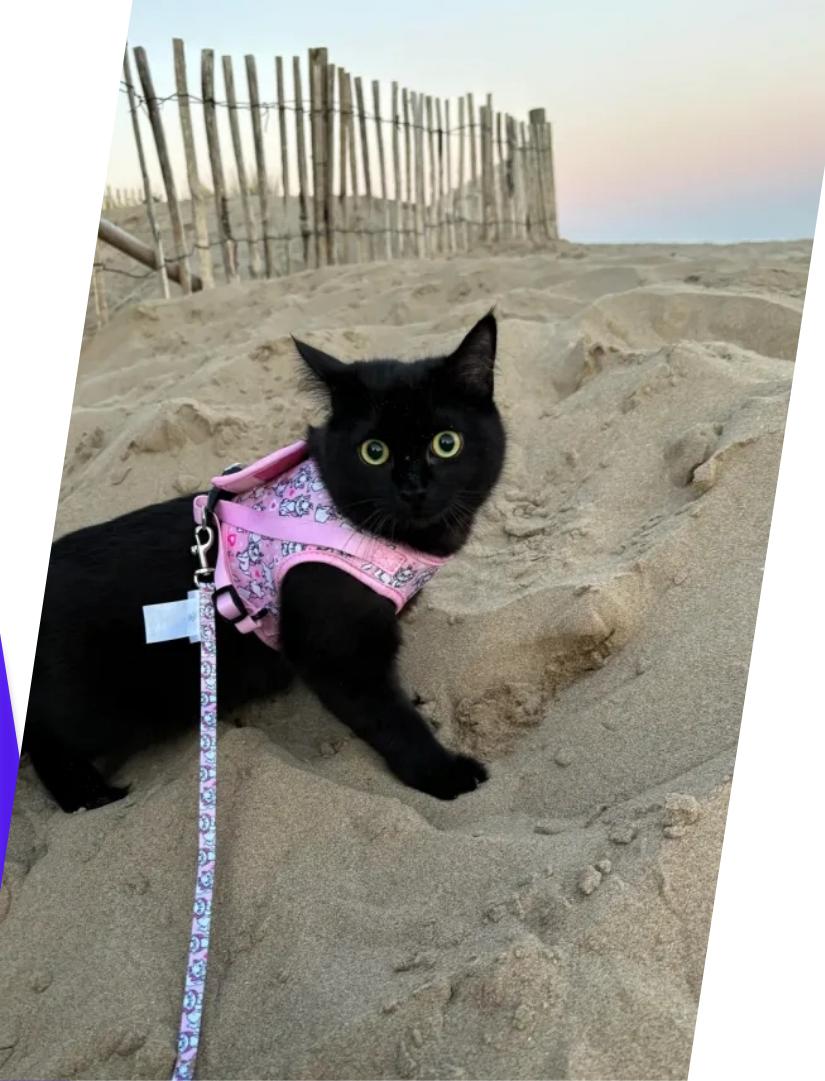
- ▶ Emulators
- ▶ Watching what the device is doing over UART/JTAG/Network

Emulators

- ▶ Emulators are great if you don't have physically access to the hardware
- ▶ They are a huge pain to setup for most IoT devices so I don't bother for most of these

Getting onto a device and seeing what's going on

- ▶ Exploring the filesystem on a live device
- ▶ Testing applications to see what they do
- ▶ Watching/modifying network traffic

A photograph of a black cat with bright green eyes, wearing a pink harness and a matching pink and white patterned leash. The cat is sitting in a sandy, dune-like environment, possibly a beach or desert area. In the background, there's a wooden fence and a clear sky with a warm sunset or sunrise glow.

UART shell

- ▶ Setup is the same as before
- ▶ We are going to use picocom to interact with this shell
- ▶ Use the credentials that you found in the copy of the firmware you already extracted
- ▶ Done

UART Shell

```
msh >dmesg
```

```
|commitid: 507a5f56d
|halgitid: 507a5f56d
|timever : Tue, 07 May 2024 16:11:56 +0800

[WRN]: [do_initcall_level:0096]: initcall: 0x43c3f9cc.
[WRN]: [do_initcall_level:0096]: initcall: 0x43c4159c.
[WRN]: [do_initcall_level:0096]: initcall: 0x43c417c8.
[WRN]: [do_initcall_level:0096]: initcall: 0x43c419c8.
[WRN]: [do_initcall_level:0096]: initcall: 0x43c422c4.
[WRN]: [do_initcall_level:0096]: initcall: 0x43c477b0.
[WRN]: [do_initcall_level:0096]: initcall: 0x43c497cc.
[INF]: [rv_to_a7_rproc_init:0080]: [AMP_INFO]Init proc ok.

[INF]: [rv_to_a7_rproc_mmap:0186]: [AMP_INFO]map pa(0x43d23c78) to va(0x43d23c78)

[INF]: [openamp_sunxi_create_rproc:0174]: [AMP_INFO]Wait master update resource_table

[WRN]: [do_initcall_level:0096]: initcall: 0x43c47338.
[WRN]: [do_initcall_level:0096]: initcall: 0x43c48ca0.
[WRN]: [do_initcall_level:0096]: initcall: 0x43c48a24.
[WRN]: [do_initcall_level:0096]: initcall: 0x43c48d84.
msh >[VIN_ERR]used1 is 0
[VIN]set clk end
[ERR]: [hal_twi_sys_pinctrl_init:1751]: [twi0] not support in sys_config
msh >
```

Other shell options

- ▶ JTAG has some ways of getting a shell
- ▶ SSH
- ▶ Telnet

A photograph of a brown tabby cat sitting on a grey carpet. To its left is a small, round emoji of an apple with a smiling face. The background shows a doorway and some furniture.

Network traffic

- ▶ How do we get it?
- ▶ What's in it?

Network setup

- ▶ Ralink WiFi adapter
- ▶ MitMProxy and hostapd
- ▶ Some fun routing rules and dnsmasq

Network setup



My Laptop with a Transparent Proxy



Analyzing network traffic

- ▶ Wireshark/Tshark



No.	Time	Source	Destination	Protocol	Length	Info
14091	317.361606	10.10.10.147	47.89.187.69	TCP	66	59656 - 443 [ACK] Seq=1 Ack=1 Win=14600 Len=0 TSval=4294941317 TSecr=931559205
14092	317.497534	10.10.10.147	47.89.187.69	TLSv1.2	509	Client Hello
14096	317.975838	10.10.10.147	47.89.187.69	TCP	66	59656 - 443 [ACK] Seq=444 Ack=1298 Win=17496 Len=0 TSval=4294941378 TSecr=931559818
14097	318.185089	10.10.10.147	47.89.187.69	TLSv1.2	141	Client Key Exchange
14099	318.187787	10.10.10.147	47.89.187.69	TLSv1.2	72	Change Cipher Spec
14101	318.190522	10.10.10.147	47.89.187.69	TLSv1.2	111	Encrypted Handshake Message
14104	318.193258	10.10.10.147	47.89.187.69	TCP	66	59656 - 443 [ACK] Seq=570 Ack=1349 Win=17496 Len=0 TSval=4294941400 TSecr=931560037
14105	318.281382	10.10.10.147	47.89.187.69	TCP	66	59656 - 443 [FIN, ACK] Seq=570 Ack=1349 Win=17496 Len=0 TSval=4294941401 TSecr=931560037
14107	318.202306	10.10.10.147	47.89.187.69	TCP	66	59656 - 443 [ACK] Seq=571 Ack=1350 Win=17496 Len=0 TSval=4294941401 TSecr=931560047
14110	320.417338	10.10.10.147	10.10.10.1	DNS	74	Standard query 0x00b6 A www.google.com
14114	322.208444	10.10.10.147	10.10.10.1	DNS	82	Standard query 0x006d A sdc-isc-us.ajcloud.net
14116	322.219668	10.10.10.147	47.89.187.69	TCP	74	59657 - 443 [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM TSval=4294941802 TSecr=0 WS=8
14118	322.219455	10.10.10.147	47.89.187.69	TCP	66	59657 - 443 [ACK] Seq=1 Ack=1 Win=14600 Len=0 TSval=4294941802 TSecr=931564056
14119	322.269483	10.10.10.147	47.89.187.69	TLSv1.2	509	Client Hello
14122	322.641686	10.10.10.147	47.89.187.69	TCP	66	59657 - 443 [ACK] Seq=444 Ack=1298 Win=17496 Len=0 TSval=4294941845 TSecr=931564482
14124	322.876068	10.10.10.147	47.89.187.69	TLSv1.2	141	Client Key Exchange
14126	322.878523	10.10.10.147	47.89.187.69	TLSv1.2	72	Change Cipher Spec
14128	322.878665	10.10.10.147	47.89.187.69	TLSv1.2	111	Encrypted Handshake Message
14131	322.881453	10.10.10.147	47.89.187.69	TCP	66	59657 - 443 [ACK] Seq=570 Ack=1349 Win=17496 Len=0 TSval=4294941869 TSecr=931564725
14132	322.882327	10.10.10.147	47.89.187.69	TCP	66	59657 - 443 [FIN, ACK] Seq=570 Ack=1349 Win=17496 Len=0 TSval=4294941869 TSecr=931564725
14134	322.885479	10.10.10.147	47.89.187.69	TCP	66	59657 - 443 [ACK] Seq=571 Ack=1350 Win=17496 Len=0 TSval=4294941869 TSecr=931564728
14139	326.887229	10.10.10.147	10.10.10.1	DNS	82	Standard query 0x006e A sdc-isc-us.ajcloud.net
14141	326.891459	10.10.10.147	47.89.187.69	TCP	74	59658 - 443 [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM TSval=4294942270 TSecr=0 WS=8
14143	326.898359	10.10.10.147	47.89.187.69	TCP	66	59658 - 443 [ACK] Seq=1 Ack=1 Win=14600 Len=0 TSval=4294942270 TSecr=931568736
14144	326.921291	10.10.10.147	10.10.10.1	DNS	74	Standard query 0x00b6 A www.google.com
14146	326.951510	10.10.10.147	47.89.187.69	TLSv1.2	509	Client Hello
14149	327.398266	10.10.10.147	47.89.187.69	TCP	66	59658 - 443 [ACK] Seq=444 Ack=1298 Win=17496 Len=0 TSval=4294942320 TSecr=931569236
14150	327.699991	10.10.10.147	47.89.187.69	TLSv1.2	141	Client Key Exchange
14152	327.709222	10.10.10.147	47.89.187.69	TLSv1.2	72	Change Cipher Spec
14154	327.701892	10.10.10.147	47.89.187.69	TLSv1.2	111	Encrypted Handshake Message
14156	327.702965	10.10.10.147	47.89.187.69	TCP	111	[TCP Spurious Retransmission] 59658 - 443 [PSH, ACK] Seq=255 Ack=1298 Win=17498 Len=45 TSval=4294942345 TSecr=93156923
14159	327.705698	10.10.10.147	47.89.187.69	TCP	66	59658 - 443 [ACK] Seq=570 Ack=1349 Win=17496 Len=0 TSval=4294942351 TSecr=931569548
14161	327.709744	10.10.10.147	47.89.187.69	TCP	66	59658 - 443 [FIN, ACK] Seq=570 Ack=1349 Win=17496 Len=0 TSval=4294942351 TSecr=931569548
14162	327.707559	10.10.10.147	47.89.187.69	TCP	66	59658 - 443 [ACK] Seq=571 Ack=1350 Win=17496 Len=0 TSval=4294942351 TSecr=931569551
14167	331.711106	10.10.10.147	10.10.10.1	DNS	82	Standard query 0x006f A sdc-isc-us.ajcloud.net
14168	331.712951	10.10.10.147	47.89.187.69	TCP	74	59659 - 443 [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM TSval=4294942752 TSecr=0 WS=8
14171	331.714156	10.10.10.147	47.89.187.69	TCP	66	59659 - 443 [ACK] Seq=1 Ack=1 Win=14600 Len=0 TSval=4294942752 TSecr=931573558
14172	331.766954	10.10.10.147	47.89.187.69	TLSv1.2	509	Client Hello
14176	332.068536	10.10.10.147	47.89.187.69	TCP	66	59659 - 443 [ACK] Seq=444 Ack=1298 Win=17496 Len=0 TSval=4294942787 TSecr=931573911
14177	332.284154	10.10.10.147	47.89.187.69	TLSv1.2	141	Client Key Exchange
14179	332.282437	10.10.10.147	47.89.187.69	TLSv1.2	72	Change Cipher Spec
14181	332.284437	10.10.10.147	47.89.187.69	TLSv1.2	111	Encrypted Handshake Message
14184	332.296671	10.10.10.147	47.89.187.69	TCP	66	59659 - 443 [ACK] Seq=570 Ack=1349 Win=17496 Len=0 TSval=4294942809 TSecr=931574131
14185	332.297489	10.10.10.147	47.89.187.69	TCP	66	59659 - 443 [FIN, ACK] Seq=570 Ack=1349 Win=17496 Len=0 TSval=4294942809 TSecr=931574131
14187	332.291178	10.10.10.147	47.89.187.69	TCP	66	59659 - 443 [ACK] Seq=571 Ack=1350 Win=17496 Len=0 TSval=4294942810 TSecr=931574133
14189	333.249848	10.10.10.147	10.10.10.1	DNS	74	Standard query 0x00b6 A www.google.com
14194	336.293922	10.10.10.147	10.10.10.1	DNS	82	Standard query 0x0070 A sdc-isc-us.ajcloud.net
14196	336.297160	10.10.10.147	47.89.187.69	TCP	74	59660 - 443 [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM TSval=4294943210 TSecr=0 WS=8
14198	336.298362	10.10.10.147	47.89.187.69	TCP	66	59660 - 443 [ACK] Seq=1 Ack=1 Win=14600 Len=0 TSval=4294943210 TSecr=931578142
14199	336.343663	10.10.10.147	47.89.187.69	TLSv1.2	509	Client Hello
14202	336.665273	10.10.10.147	47.89.187.69	TCP	66	59660 - 443 [ACK] Seq=444 Ack=1298 Win=17496 Len=0 TSval=4294943250 TSecr=931578535
14203	336.895681	10.10.10.147	47.89.187.69	TLSv1.2	141	Client Key Exchange
14205	336.896625	10.10.10.147	47.89.187.69	TLSv1.2	72	Change Cipher Spec
14207	336.897996	10.10.10.147	47.89.187.69	TLSv1.2	111	Encrypted Handshake Message
14210	336.898977	10.10.10.147	47.89.187.69	TCP	66	59660 - 443 [ACK] Seq=570 Ack=1349 Win=17496 Len=0 TSval=4294943271 TSecr=931578743
14211	336.901055	10.10.10.147	47.89.187.69	TCP	66	59660 - 443 [FIN, ACK] Seq=570 Ack=1349 Win=17496 Len=0 TSval=4294943271 TSecr=931578743
14213	336.902742	10.10.10.147	47.89.187.69	TCP	66	59660 - 443 [ACK] Seq=571 Ack=1350 Win=17496 Len=0 TSval=4294943271 TSecr=931578747
14217	339.924428	10.10.10.147	10.10.10.1	DNS	74	Standard query 0x00b6 A www.google.com

```
Frame 14139: 82 bytes on wire (656 bits), 82 bytes captured (656 bits)
Ethernet II, Src: AlteBeamo_b5:93:83 (00:31:ab:b5:93:83), Dst: ASRockIn_44:65:54 (00:25:22:44:65:54)
Internet Protocol Version 4, Src: 10.10.10.147, Dst: 10.10.10.1
>User Datagram Protocol, Src Port: 42255, Dst Port: 53
  Domain Name System (query)
```

```
0000 00 25 22 44 65 54 90 31 b4 b5 93 88 00 45 00 "%DeT1 K - E  
0010 00 44 9e 3d 49 00 40 11 73 c4 0a 0a 93 00 0a 00 D = 0 . s  
0020 0a 01 a5 0f 00 35 00 30 05 5c 00 6e 01 00 00 01 ..... 5 0 \n  
0030 00 00 00 00 00 00 0a 73 64 63 2d 69 73 63 2d 75 ..... s dc-isc-u  
0040 73 07 61 6a 63 6c 6f 75 64 03 6e 65 74 00 00 01 s ajclou d net
```



SSL
certificate
abuse

OzSec 2025



Lack of certificate pinning

- ▶ Many devices don't have certificate pinning
- ▶ This allows you to MitM TLS connections, not just unencrypted connections

Ways IoT devices can be better secured

- ▶ Locking down the bootloader
- ▶ Stop shipping things in debug mode
- ▶ Encrypted flash
- ▶ Not shipping a MVP
- ▶ Not using really dumb password
- ▶ Verifying OTA updates
- ▶ Pinning SSL certificates

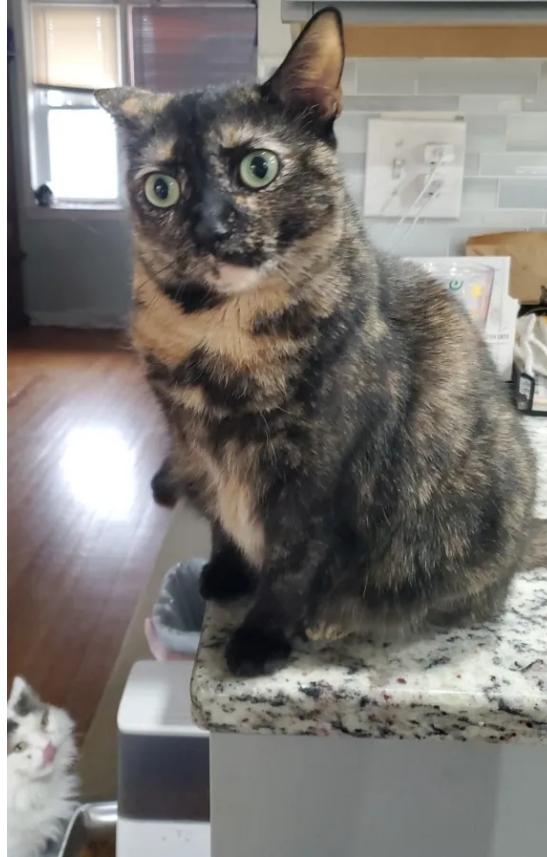
Those all cost money



OzSec 2025

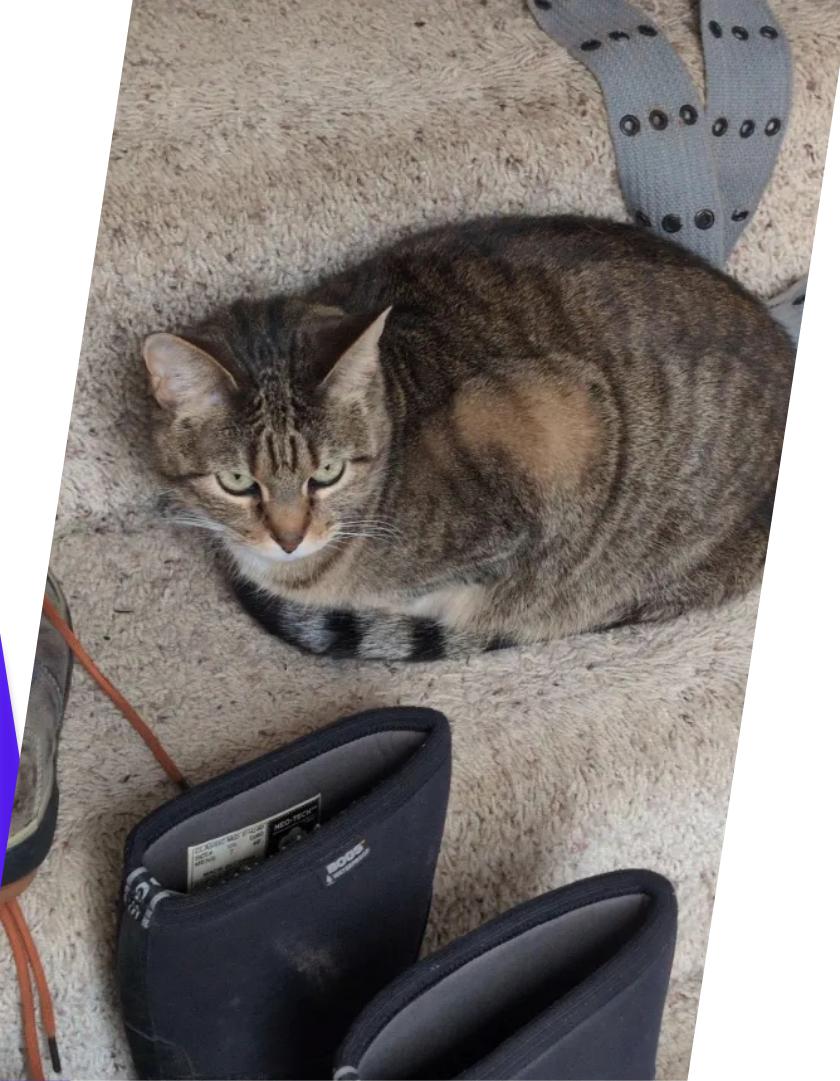


Possible impacts



OzSec 2025

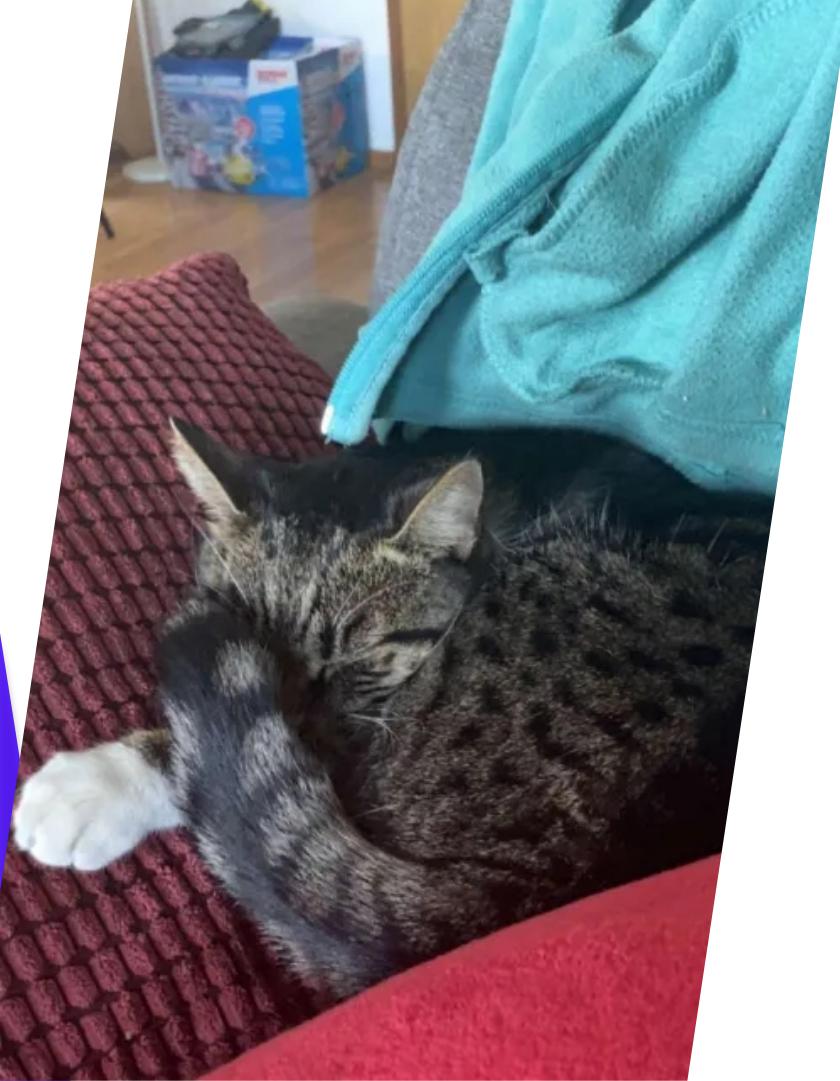


A photograph of a brown tabby cat sitting on a light-colored carpet. The cat is looking towards the camera. In the foreground, the top of a pair of black rubber boots is visible. A grey strap with circular holes lies across the carpet behind the cat.

Initial foothold

- ▶ Routers
- ▶ IP cameras

PUT IoT DEVICES ON THEIR OWN VLAN



Botnets

- ▶ Mirai
- ▶ Bashlite
- ▶ Aisuru (apparently behind the attacks this week)
- ▶ Qbot
- ▶ Reaper



Questions?

OzSec 2025





Thanks for listening

- ▶ Thank you for listening to me talk about this stuff, I hope you all enjoyed.

You can find me at
www.powershell.zip or
at SeckC every month

OzSec 2026

- Thank you for Attending!!
- Looking forward to next year already
- Pre-Register for OzSec2026
 - <https://ozsecurity.org/>