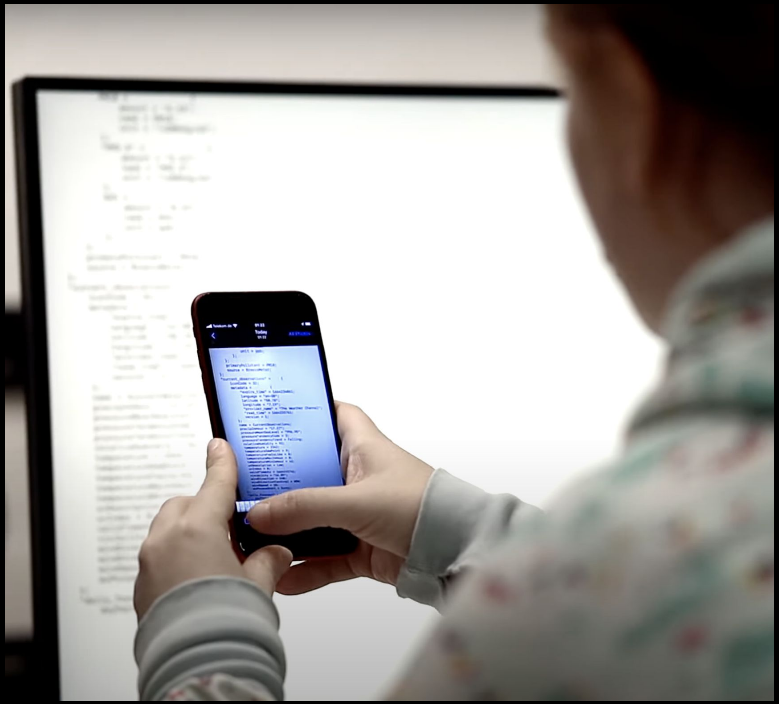
Hacking Closed-Source

The Power of Reverse Engineering Real-World Products

Jiska Classen
Secure Mobile Networking Lab - SEEMOO
TU Darmstadt, Germany

HACKADAY berlin 2023

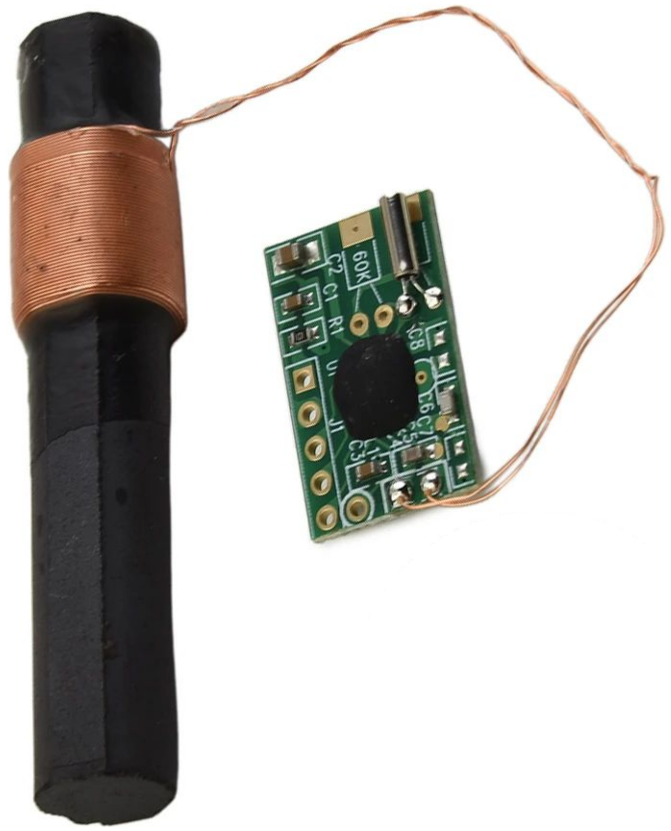
Me age 14:
Open-source software
evangelist.



Me now:
Open the source software
evangelist.

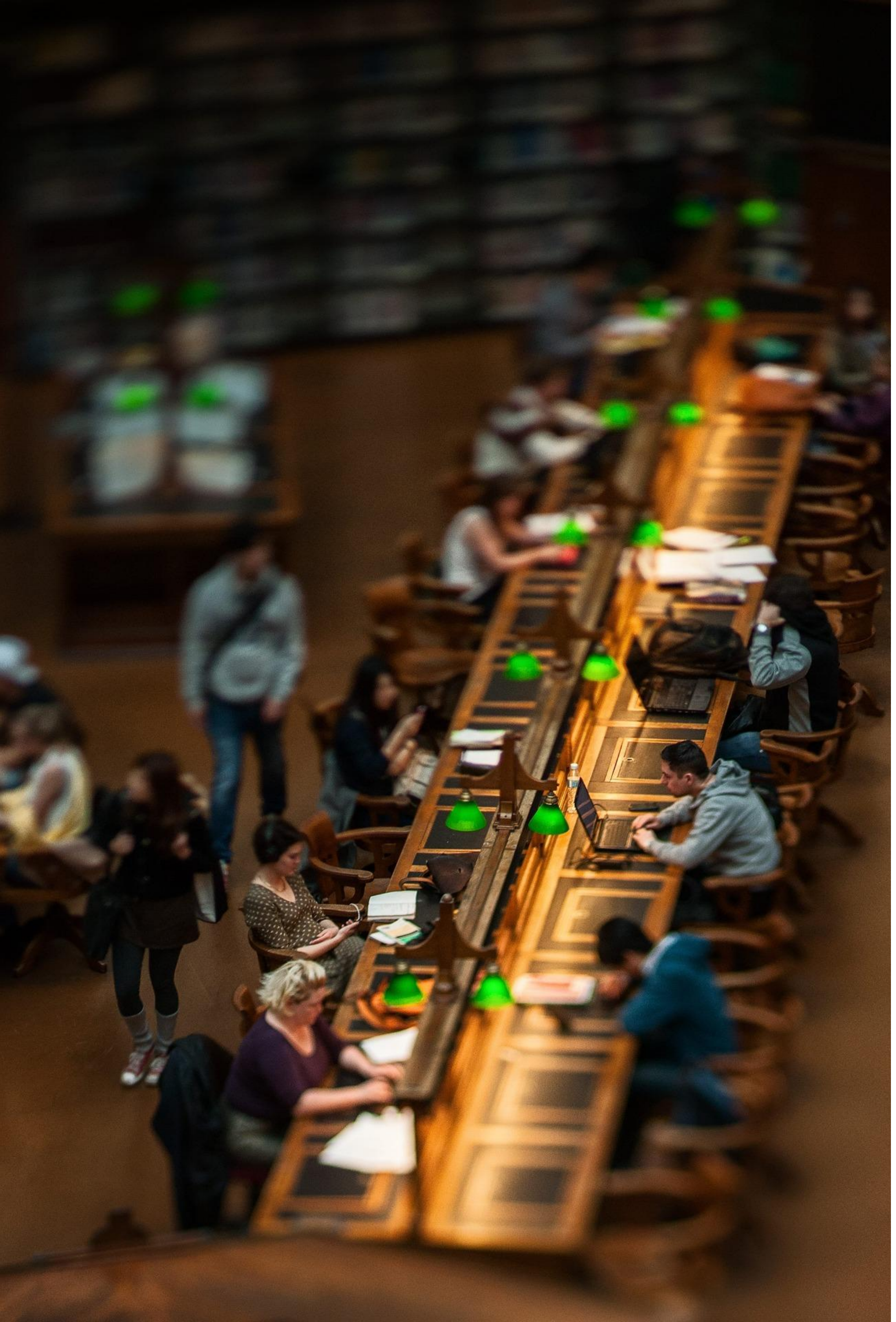
Time
00:08 53
x 003
61
km/h

School 👩‍🎓



Programming a DCF77 receiver in Assembly.

C is hard and has so many different instructions.



Studying 👩‍🎓

Nothing I learned at the Linux User Group is relevant for my studies.

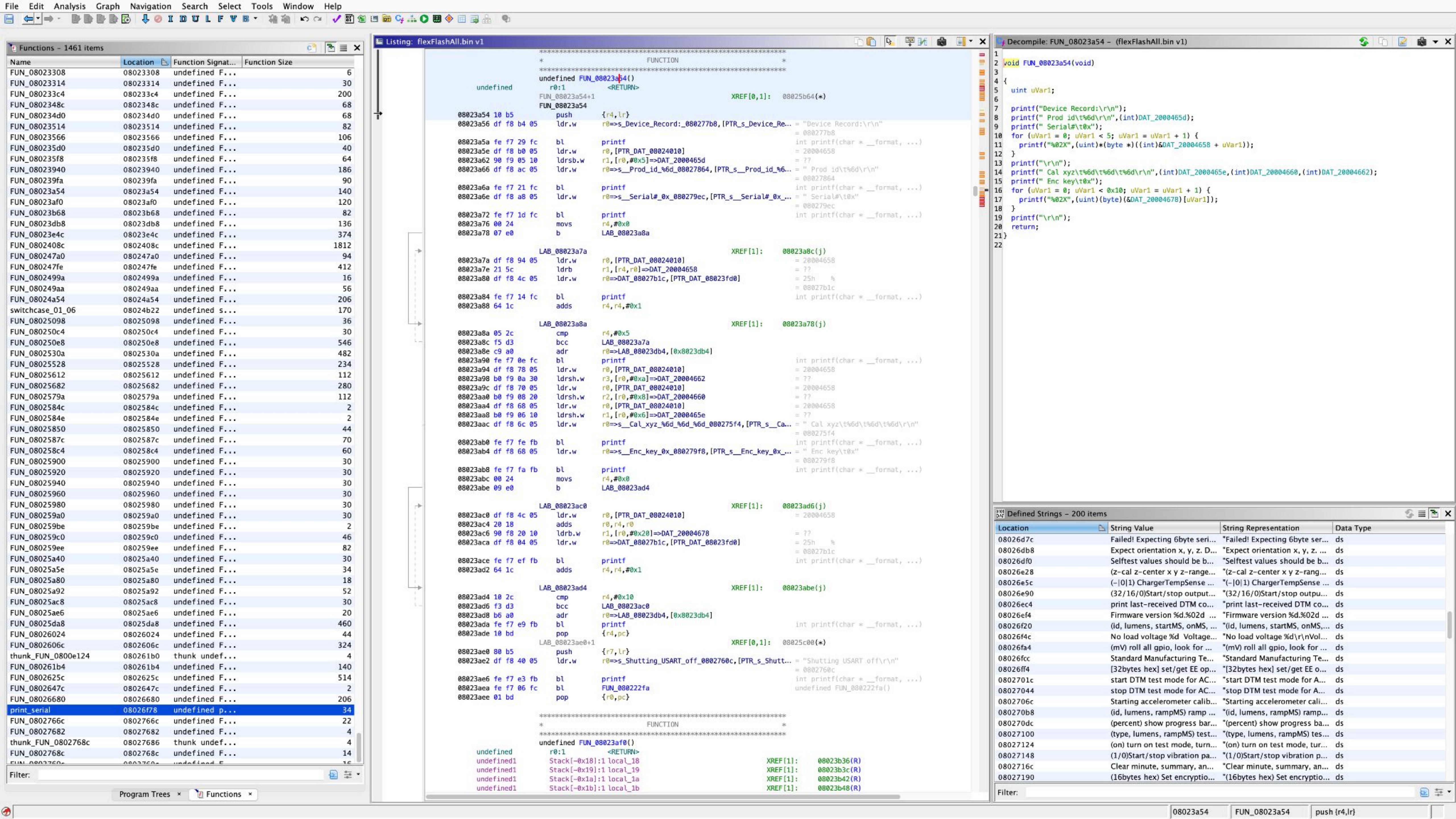
What am I even studying?

Is anything of this relevant to security?

That M.Sc. IT Security sounds interesting…

I still know nothing, maybe I should stay for a PhD?

“We have that fitness tracker firmware and nobody has the time to look into it.”



Functions - 1461 items

Name	Location	Function Signat...	Function Size
FUN_08023308	08023308	undefined F...	6
FUN_08023314	08023314	undefined F...	30
FUN_080233c4	080233c4	undefined F...	200
FUN_0802348c	0802348c	undefined F...	68
FUN_080234d0	080234d0	undefined F...	68
FUN_08023514	08023514	undefined F...	82
FUN_08023566	08023566	undefined F...	106
FUN_080235d0	080235d0	undefined F...	40
FUN_080235f8	080235f8	undefined F...	64
FUN_08023940	08023940	undefined F...	186
FUN_080239fa	080239fa	undefined F...	90
FUN_08023a54	08023a54	undefined F...	140
FUN_08023af0	08023af0	undefined F...	120
FUN_08023b68	08023b68	undefined F...	82
FUN_08023db8	08023db8	undefined F...	136
FUN_08023e4c	08023e4c	undefined F...	374
FUN_0802408c	0802408c	undefined F...	1812
FUN_080247a0	080247a0	undefined F...	94
FUN_080247fe	080247fe	undefined F...	412
FUN_0802499a	0802499a	undefined F...	16
FUN_080249aa	080249aa	undefined F...	56
FUN_08024a54	08024a54	undefined F...	206
switchcase_01_06	08024b22	undefined s...	170
FUN_08025098	08025098	undefined F...	36
FUN_080250c4	080250c4	undefined F...	30
FUN_080250e8	080250e8	undefined F...	546
FUN_0802530a	0802530a	undefined F...	482
FUN_08025528	08025528	undefined F...	234
FUN_08025612	08025612	undefined F...	112
FUN_08025682	08025682	undefined F...	280
FUN_0802579a	0802579a	undefined F...	112
FUN_0802584c	0802584c	undefined F...	2
FUN_0802584e	0802584e	undefined F...	2
FUN_08025850	08025850	undefined F...	44
FUN_0802587c	0802587c	undefined F...	70
FUN_080258c4	080258c4	undefined F...	60
FUN_08025900	08025900	undefined F...	30
FUN_08025920	08025920	undefined F...	30
FUN_08025940	08025940	undefined F...	30
FUN_08025960	08025960	undefined F...	30
FUN_08025980	08025980	undefined F...	30
FUN_080259a0	080259a0	undefined F...	30
FUN_080259be	080259be	undefined F...	2
FUN_080259c0	080259c0	undefined F...	46
FUN_080259ee	080259ee	undefined F...	82
FUN_08025a40	08025a40	undefined F...	30
FUN_08025a5e	08025a5e	undefined F...	34
FUN_08025a80	08025a80	undefined F...	18
FUN_08025a92	08025a92	undefined F...	52
FUN_08025ac8	08025ac8	undefined F...	30
FUN_08025ae6	08025ae6	undefined F...	20
FUN_08025da8	08025da8	undefined F...	460
FUN_08026024	08026024	undefined F...	44
FUN_0802606c	0802606c	undefined F...	324
thunk_FUN_0800e124	080261b0	thunk undef...	4
FUN_080261b4	080261b4	undefined F...	140
FUN_0802625c	0802625c	undefined F...	514
FUN_0802647c	0802647c	undefined F...	2
FUN_08026680	08026680	undefined F...	206
print_serial	08026f78	undefined p...	34
FUN_0802766c	0802766c	undefined F...	22
FUN_08027682	08027682	undefined F...	4
thunk_FUN_0802768c	08027686	thunk undef...	4
FUN_0802768c	0802768c	undefined F...	14

Filter:

Listing: flexFlashAll.bin v1

```
                     **************************************************************
                     *                          FUNCTION                          *
                     **************************************************************
                     undefined FUN_08023a54()
     undefined         r0:1           <RETURN>
                     FUN_08023a54+1                                XREF[0,1]:   08025b64(*)
                     FUN_08023a54
08023a54 10 b5           push       {r4,lr}
08023a56 df f8 b4 05     ldr.w      r0=>s_Device_Record:_080277b8,[PTR_s_Device_Re... = "Device Record:\r\n"
                                                                                  = 080277b8
08023a5a fe f7 29 fc     bl         printf                                         int printf(char * __format, ...)
08023a5e df f8 b0 05     ldr.w      r0,[PTR_DAT_08024010]                          = 20004658
08023a62 90 f9 05 10     ldrsb.w    r1,[r0,#0x5]=>DAT_2000465d                     = ??
08023a66 df f8 ac 05     ldr.w      r0=>s__Prod_id_%6d_08027864,[PTR_s__Prod_id_%6... = " Prod id\t%6d\r\n"
                                                                                  = 08027864
08023a6a fe f7 21 fc     bl         printf                                         int printf(char * __format, ...)
08023a6e df f8 a8 05     ldr.w      r0=>s__Serial#_0x_080279ec,[PTR_s__Serial#_0x_... = " Serial#\t0x"
                                                                                  = 080279ec
08023a72 fe f7 1d fc     bl         printf                                         int printf(char * __format, ...)
08023a76 00 24           movs       r4,#0x0
08023a78 07 e0           b          LAB_08023a8a

                     LAB_08023a7a                                  XREF[1]:     08023a8c(j)
08023a7a df f8 94 05     ldr.w      r0,[PTR_DAT_08024010]                          = 20004658
08023a7e 21 5c           ldrb       r1,[r4,r0]=>DAT_20004658                       = ??
08023a80 df f8 4c 05     ldr.w      r0=>DAT_08027b1c,[PTR_DAT_08023fd0]            = 25h    %
                                                                                  = 08027b1c
08023a84 fe f7 14 fc     bl         printf                                         int printf(char * __format, ...)
08023a88 64 1c           adds       r4,r4,#0x1

                     LAB_08023a8a                                  XREF[1]:     08023a78(j)
08023a8a 05 2c           cmp        r4,#0x5
08023a8c f5 d3           bcc        LAB_08023a7a
08023a8e c9 a0           adr        r0=>LAB_08023db4,[0x8023db4]
08023a90 fe f7 0e fc     bl         printf                                         int printf(char * __format, ...)
08023a94 df f8 78 05     ldr.w      r0,[PTR_DAT_08024010]                          = 20004658
08023a98 b0 f9 0a 30     ldrsh.w    r3,[r0,#0xa]=>DAT_20004662                     = ??
08023a9c df f8 70 05     ldr.w      r0,[PTR_DAT_08024010]                          = 20004658
08023aa0 b0 f9 08 20     ldrsh.w    r2,[r0,#0x8]=>DAT_20004660                     = ??
08023aa4 df f8 68 05     ldr.w      r0,[PTR_DAT_08024010]                          = 20004658
08023aa8 b0 f9 06 10     ldrsh.w    r1,[r0,#0x6]=>DAT_2000465e                     = ??
08023aac df f8 6c 05     ldr.w      r0=>s__Cal_xyz_%6d_%6d_%6d_080275f4,[PTR_s__Ca... = " Cal xyz\t%6d\t%6d\t%6d\r\n"
                                                                                  = 080275f4
08023ab0 fe f7 fe fb     bl         printf                                         int printf(char * __format, ...)
08023ab4 df f8 68 05     ldr.w      r0=>s__Enc_key_0x_080279f8,[PTR_s__Enc_key_0x_... = " Enc key\t0x"
                                                                                  = 080279f8
08023ab8 fe f7 fa fb     bl         printf                                         int printf(char * __format, ...)
08023abc 00 24           movs       r4,#0x0
08023abe 09 e0           b          LAB_08023ad4

                     LAB_08023ac0                                  XREF[1]:     08023ad6(j)
08023ac0 df f8 4c 05     ldr.w      r0,[PTR_DAT_08024010]                          = 20004658
08023ac4 20 18           adds       r0,r4,r0
08023ac6 90 f8 20 10     ldrb.w     r1,[r0,#0x20]=>DAT_20004678                    = ??
08023aca df f8 04 05     ldr.w      r0=>DAT_08027b1c,[PTR_DAT_08023fd0]            = 25h    %
                                                                                  = 08027b1c
08023ace fe f7 ef fb     bl         printf                                         int printf(char * __format, ...)
08023ad2 64 1c           adds       r4,r4,#0x1

                     LAB_08023ad4                                  XREF[1]:     08023abe(j)
08023ad4 10 2c           cmp        r4,#0x10
08023ad6 f3 d3           bcc        LAB_08023ac0
08023ad8 b6 a0           adr        r0=>LAB_08023db4,[0x8023db4]
08023ada fe f7 e9 fb     bl         printf                                         int printf(char * __format, ...)
08023ade 10 bd           pop        {r4,pc}
                     LAB_08023ae0+1                                XREF[0,1]:   08025c00(*)
08023ae0 80 b5           push       {r7,lr}
08023ae2 df f8 40 05     ldr.w      r0=>s_Shutting_USART_off_0802760c,[PTR_s_Shutt... = "Shutting USART off\r\n"
                                                                                  = 0802760c
08023ae6 fe f7 e3 fb     bl         printf                                         int printf(char * __format, ...)
08023aea fe f7 06 fc     bl         FUN_080222fa                                   undefined FUN_080222fa()
08023aee 01 bd           pop        {r0,pc}

                     **************************************************************
                     *                          FUNCTION                          *
                     **************************************************************
                     undefined FUN_08023af0()
     undefined         r0:1           <RETURN>
     undefined1        Stack[-0x18]:1 local_18                        XREF[1]:     08023b36(R)
     undefined1        Stack[-0x19]:1 local_19                        XREF[1]:     08023b3c(R)
     undefined1        Stack[-0x1a]:1 local_1a                        XREF[1]:     08023b42(R)
     undefined1        Stack[-0x1b]:1 local_1b                        XREF[1]:     08023b48(R)
```

Decompile: FUN_08023a54 - (flexFlashAll.bin v1)

```c

void FUN_08023a54(void)

{
  uint uVar1;

  printf("Device Record:\r\n");
  printf(" Prod id\t%6d\r\n",(int)DAT_2000465d);
  printf(" Serial#\t0x");
  for (uVar1 = 0; uVar1 < 5; uVar1 = uVar1 + 1) {
    printf("%02X",(uint)*(byte *)((int)&DAT_20004658 + uVar1));
  }
  printf("\r\n");
  printf(" Cal xyz\t%6d\t%6d\t%6d\r\n",(int)DAT_2000465e,(int)DAT_20004660,(int)DAT_20004662);
  printf(" Enc key\t0x");
  for (uVar1 = 0; uVar1 < 0x10; uVar1 = uVar1 + 1) {
    printf("%02X",(uint)(byte)(&DAT_20004678)[uVar1]);
  }
  printf("\r\n");
  return;
}

```

Defined Strings - 200 items

Location	String Value	String Representation	Data Type
08026d7c	Failed! Expecting 6byte seri...	"Failed! Expecting 6byte ser...	ds
08026db8	Expect orientation x, y, z. D...	"Expect orientation x, y, z. ...	ds
08026df0	Selftest values should be b...	"Selftest values should be b...	ds
08026e28	(z-cal z-center x y z-range...	"(z-cal z-center x y z-rang...	ds
08026e5c	(-|0|1) ChargerTempSense ...	"(-|0|1) ChargerTempSense ...	ds
08026e90	(32/16/0)Start/stop output...	"(32/16/0)Start/stop outpu...	ds
08026ec4	print last-received DTM co...	"print last-received DTM co...	ds
08026ef4	Firmware version %d.%02d ...	"Firmware version %d.%02d ...	ds
08026f20	(id, lumens, startMS, onMS, ...	"(id, lumens, startMS, onMS,...	ds
08026f4c	No load voltage %d Voltage...	"No load voltage %d\r\nVol...	ds
08026fa4	(mV) roll all gpio, look for ...	"(mV) roll all gpio, look for ...	ds
08026fcc	Standard Manufacturing Te...	"Standard Manufacturing Te...	ds
08026ff4	[32bytes hex] set/get EE op...	"[32bytes hex] set/get EE o...	ds
0802701c	start DTM test mode for AC...	"start DTM test mode for A...	ds
08027044	stop DTM test mode for AC...	"stop DTM test mode for A...	ds
0802706c	Starting accelerometer calib...	"Starting accelerometer cali...	ds
080270b8	(id, lumens, rampMS) ramp ...	"(id, lumens, rampMS) ramp...	ds
080270dc	(percent) show progress bar...	"(percent) show progress ba...	ds
08027100	(type, lumens, rampMS) test...	"(type, lumens, rampMS) tes...	ds
08027124	(on) turn on test mode, turn...	"(on) turn on test mode, tur...	ds
08027148	(1/0)Start/stop vibration pa...	"(1/0)Start/stop vibration p...	ds
0802716c	Clear minute, summary, an...	"Clear minute, summary, an...	ds
08027190	(16bytes hex) Set encryptio...	"(16bytes hex) Set encryptio...	ds

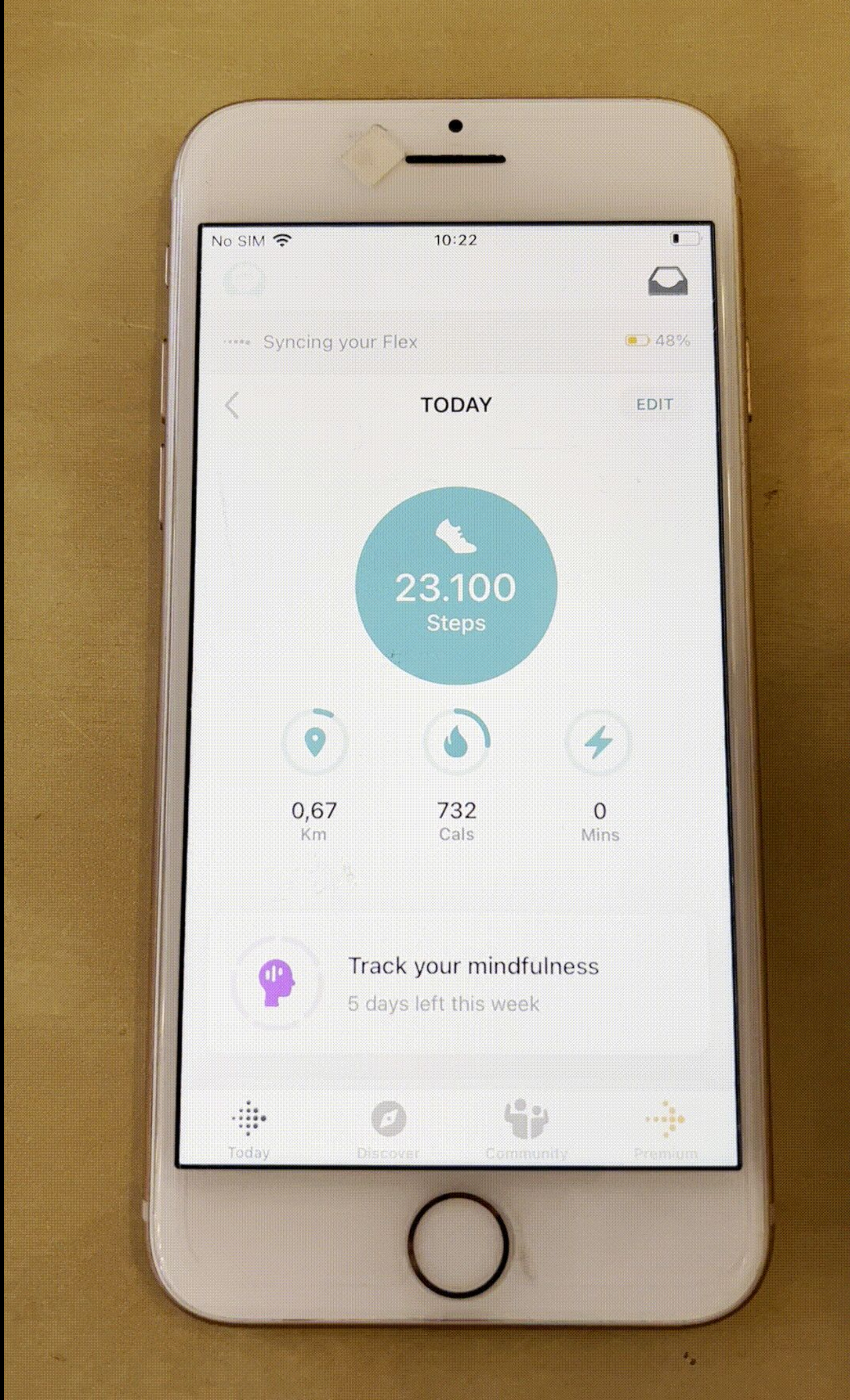
Filter:

08023a54 | FUN_08023a54 | push {r4,lr}

staring intensifies

Years of learning random things
suddenly make sense.

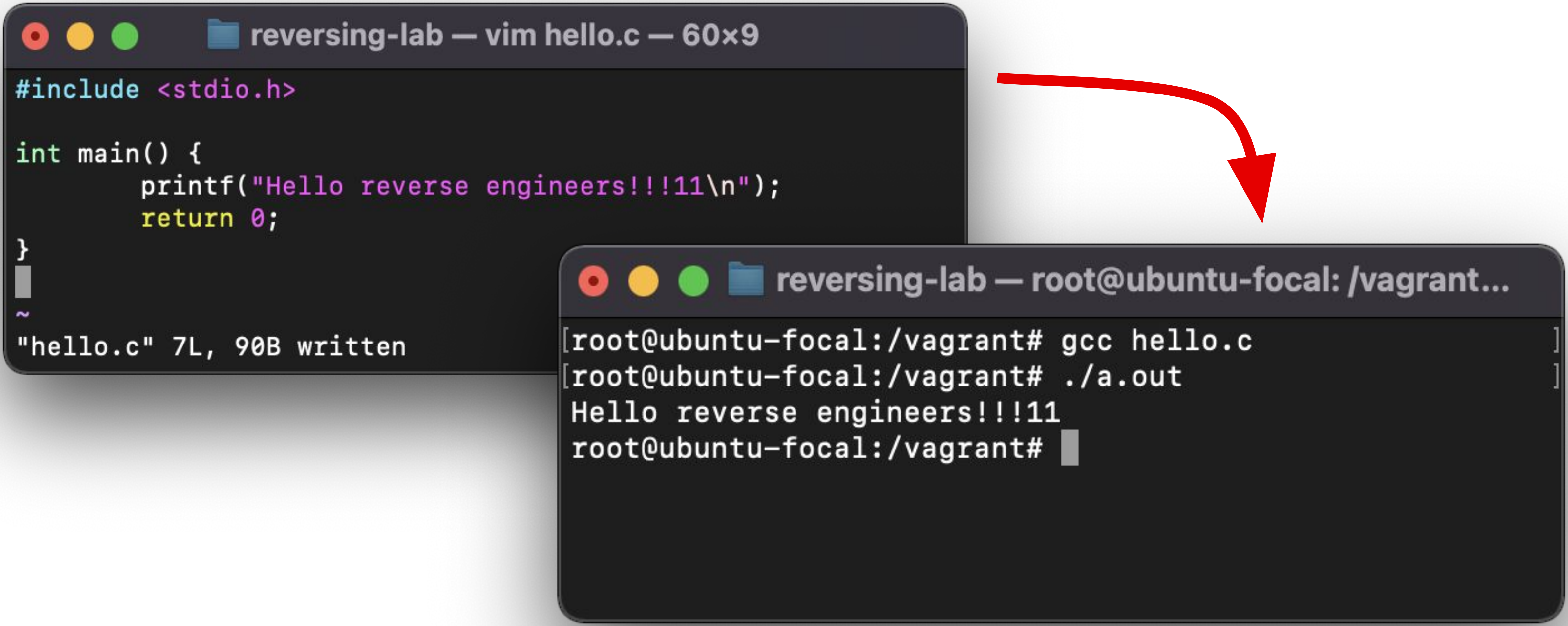
```c
#pragma NEXMON targetregion "patch"

#include <firmware_version.h>
#include <patcher.h>
#include <wrapper.h>
#include "stm32.h"

int hook_get_steps() {
   int steps = *((int *) 0x20003B54);
   steps = steps * 100;
   return steps;
}

__attribute__((at(0x8014304, "", CHIP_VER_FITBIT, FW_VER_FITBIT)))
BPatch(hook_get_steps, hook_get_steps);
```



Everything is open-source now.

◀◀ REW

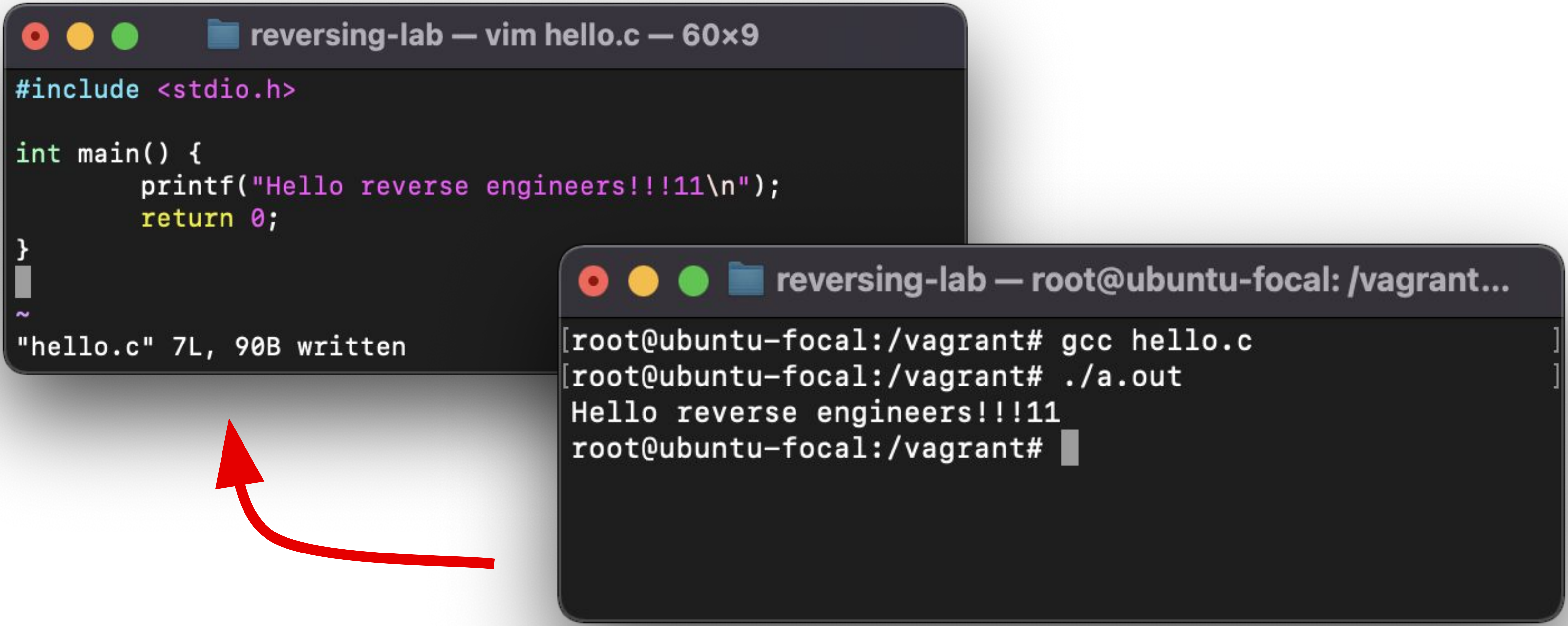
Forward Engineering 🦥


reversing-lab — vim hello.c — 60×9
reversing-lab — root@ubuntu-focal: /vagrant...

```
#include <stdio.h>

int main() {
        printf("Hello reverse engineers!!!11\n");
        return 0;
}

~
"hello.c" 7L, 90B written
```

```
root@ubuntu-focal:/vagrant# gcc hello.c
root@ubuntu-focal:/vagrant# ./a.out
Hello reverse engineers!!!11
root@ubuntu-focal:/vagrant#
```

Reverse Engineering 🤓



reversing-lab — vim hello.c — 60×9

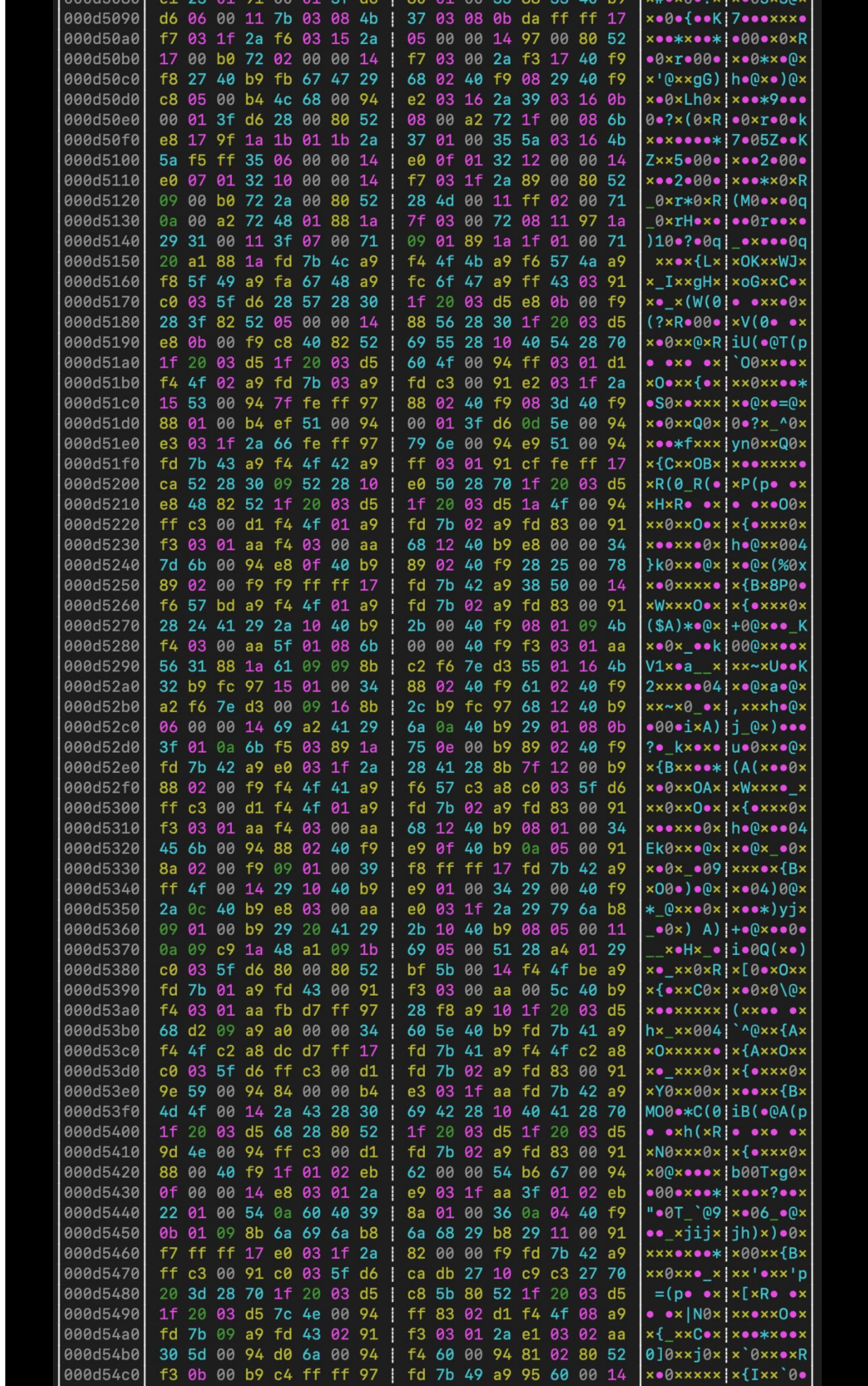
```c
#include <stdio.h>

int main() {
        printf("Hello reverse engineers!!!11\n");
        return 0;
}

~
"hello.c" 7L, 90B written
```

reversing-lab — root@ubuntu-focal: /vagrant...

```
root@ubuntu-focal:/vagrant# gcc hello.c
root@ubuntu-focal:/vagrant# ./a.out
Hello reverse engineers!!!11
root@ubuntu-focal:/vagrant#
```

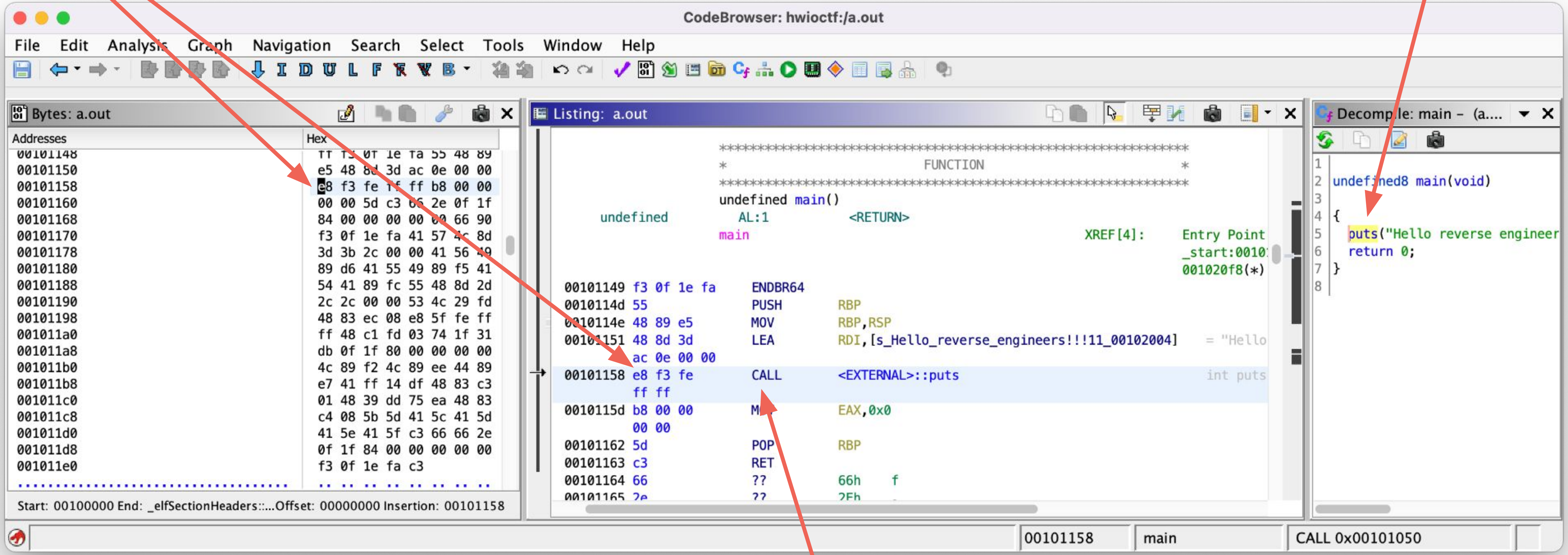
Why reverse engineer something?

- Most software ships as binary without source code.
- Even if you have source code, libraries and system components used by a program might be binary-only.
- When analyzing real-world software, reverse engineering is indispensable.



```
│000d5090│ d6 06 00 11 7b 03 08 4b ┊ 37 03 08 0b da ff ff 17 │×•0•{••K┊7•••×××•│
│000d50a0│ f7 03 1f 2a f6 03 15 2a ┊ 05 00 00 14 97 00 80 52 │×••*×••*┊•00•×0×R│
│000d50b0│ 17 00 b0 72 02 00 00 14 ┊ f7 03 00 2a f3 17 40 f9 │•0×r•00•┊×•0*×•@×│
│000d50c0│ f8 27 40 b9 fb 67 47 29 ┊ 68 02 40 f9 08 29 40 f9 │×'@××gG)┊h•@×•)@×│
│000d50d0│ c8 05 00 b4 4c 68 00 94 ┊ e2 03 16 2a 39 03 16 0b │×•0×Lh0×┊×••*9•••│
│000d50e0│ 00 01 3f d6 28 00 80 52 ┊ 08 00 a2 72 1f 00 08 6b │0•?×(0×R┊•0×r•0•k│
│000d50f0│ e8 17 9f 1a 1b 01 1b 2a ┊ 37 01 00 35 5a 03 16 4b │×•×••••*┊7•05Z••K│
│000d5100│ 5a f5 ff 35 06 00 00 14 ┊ e0 0f 01 32 12 00 00 14 │Z××5•00•┊×••2•00•│
│000d5110│ e0 07 01 32 10 00 00 14 ┊ f7 03 1f 2a 89 00 80 52 │×••2•00•┊×••*×0×R│
│000d5120│ 09 00 b0 72 2a 00 80 52 ┊ 28 4d 00 11 ff 02 00 71 │_0×r*0×R┊(M0•×•0q│
│000d5130│ 0a 00 a2 72 48 01 88 1a ┊ 7f 03 00 72 08 11 97 1a │_0×rH•×•┊••0r••×•│
│000d5140│ 29 31 00 11 3f 07 00 71 ┊ 09 01 89 1a 1f 01 00 71 │)10•?•0q┊_•×•••0q│
│000d5150│ 20 a1 88 1a fd 7b 4c a9 ┊ f4 4f 4b a9 f6 57 4a a9 │ ××•×{L×┊×OK××WJ×│
│000d5160│ f8 5f 49 a9 fa 67 48 a9 ┊ fc 6f 47 a9 ff 43 03 91 │×_I××gH×┊×oG××C•×│
│000d5170│ c0 03 5f d6 28 57 28 30 ┊ 1f 20 03 d5 e8 0b 00 f9 │×•_×(W(0┊• •××•0×│
│000d5180│ 28 3f 82 52 05 00 00 14 ┊ 88 56 28 30 1f 20 03 d5 │(?×R•00•┊×V(0• •×│
│000d5190│ e8 0b 00 f9 c8 40 82 52 ┊ 69 55 28 10 40 54 28 70 │×•0××@×R┊iU(•@T(p│
│000d51a0│ 1f 20 03 d5 1f 20 03 d5 ┊ 60 4f 00 94 ff 03 01 d1 │• •×• •×┊`O0××••×│
│000d51b0│ f4 4f 02 a9 fd 7b 03 a9 ┊ fd c3 00 91 e2 03 1f 2a │×O•××{•×┊××0××••*│
│000d51c0│ 15 53 00 94 7f fe ff 97 ┊ 88 02 40 f9 08 3d 40 f9 │•S0×•×××┊×•@×•=@×│
│000d51d0│ 88 01 00 b4 ef 51 00 94 ┊ 00 01 3f d6 0d 5e 00 94 │×•0××Q0×┊0•?×_^0×│
│000d51e0│ e3 03 1f 2a 66 fe ff 97 ┊ 79 6e 00 94 e9 51 00 94 │×••*f×××┊yn0××Q0×│
│000d51f0│ fd 7b 43 a9 f4 4f 42 a9 ┊ ff 03 01 91 cf fe ff 17 │×{C××OB×┊×••××××•│
│000d5200│ ca 52 28 30 09 52 28 10 ┊ e0 50 28 70 1f 20 03 d5 │×R(0_R(•┊×P(p• •×│
│000d5210│ e8 48 82 52 1f 20 03 d5 ┊ 1f 20 03 d5 1a 4f 00 94 │×H×R• •×┊• •×•O0×│
│000d5220│ ff c3 00 d1 f4 4f 01 a9 ┊ fd 7b 02 a9 fd 83 00 91 │××0××O•×┊×{•×××0×│
│000d5230│ f3 03 01 aa f4 03 00 aa ┊ 68 12 40 b9 e8 00 00 34 │×••××•0×┊h•@××004│
│000d5240│ 7d 6b 00 94 e8 0f 40 b9 ┊ 89 02 40 f9 28 25 00 78 │}k0××•@×┊×•@×(%0x│
│000d5250│ 89 02 00 f9 f9 ff ff 17 ┊ fd 7b 42 a9 38 50 00 14 │×•0××××•┊×{B×8P0•│
│000d5260│ f6 57 bd a9 f4 4f 01 a9 ┊ fd 7b 02 a9 fd 83 00 91 │×W×××O•×┊×{•×××0×│
│000d5270│ 28 24 41 29 2a 10 40 b9 ┊ 2b 00 40 f9 08 01 09 4b │($A)*•@×┊+0@×••_K│
│000d5280│ f4 03 00 aa 5f 01 08 6b ┊ 00 00 40 f9 f3 03 01 aa │×•0×_••k┊00@××••×│
│000d5290│ 56 31 88 1a 61 09 09 8b ┊ c2 f6 7e d3 55 01 16 4b │V1×•a__×┊××~×U••K│
│000d52a0│ 32 b9 fc 97 15 01 00 34 ┊ 88 02 40 f9 61 02 40 f9 │2×××••04┊×•@×a•@×│
│000d52b0│ a2 f6 7e d3 00 09 16 8b ┊ 2c b9 fc 97 68 12 40 b9 │××~×0_•×┊,×××h•@×│
│000d52c0│ 06 00 00 14 69 a2 41 29 ┊ 6a 0a 40 b9 29 01 08 0b │•00•i×A)┊j_@×)•••│
│000d52d0│ 3f 01 0a 6b f5 03 89 1a ┊ 75 0e 00 b9 89 02 40 f9 │?•_k×•×•┊u•0××•@×│
│000d52e0│ fd 7b 42 a9 e0 03 1f 2a ┊ 28 41 28 8b 7f 12 00 b9 │×{B××••*┊(A(×••0×│
│000d52f0│ 88 02 00 f9 f4 4f 41 a9 ┊ f6 57 c3 a8 c0 03 5f d6 │×•0××OA×┊×W×××•_×│
│000d5300│ ff c3 00 d1 f4 4f 01 a9 ┊ fd 7b 02 a9 fd 83 00 91 │××0××O•×┊×{•×××0×│
│000d5310│ f3 03 01 aa f4 03 00 aa ┊ 68 12 40 b9 08 01 00 34 │×••××•0×┊h•@×••04│
│000d5320│ 45 6b 00 94 88 02 40 f9 ┊ e9 0f 40 b9 0a 05 00 91 │Ek0××•@×┊×•@×_•0×│
│000d5330│ 8a 02 00 f9 09 01 00 39 ┊ f8 ff ff 17 fd 7b 42 a9 │×•0×_•09┊×××•×{B×│
│000d5340│ ff 4f 00 14 29 10 40 b9 ┊ e9 01 00 34 29 00 40 f9 │×O0•)•@×┊×•04)0@×│
│000d5350│ 2a 0c 40 b9 e8 03 00 aa ┊ e0 03 1f 2a 29 79 6a b8 │*_@××•0×┊×••*)yj×│
│000d5360│ 09 01 00 b9 29 20 41 29 ┊ 2b 10 40 b9 08 05 00 11 │_•0×) A)┊+•@×••0•│
│000d5370│ 0a 09 c9 1a 48 a1 09 1b ┊ 69 05 00 51 28 a4 01 29 │__×•H×_•┊i•0Q(×•)│
│000d5380│ c0 03 5f d6 80 00 80 52 ┊ bf 5b 00 14 f4 4f be a9 │×•_××0×R┊×[0•×O××│
│000d5390│ fd 7b 01 a9 fd 43 00 91 ┊ f3 03 00 aa 00 5c 40 b9 │×{•××C0×┊×•0×0\@×│
│000d53a0│ f4 03 01 aa fb d7 ff 97 ┊ 28 f8 a9 10 1f 20 03 d5 │×••×××××┊(××•• •×│
│000d53b0│ 68 d2 09 a9 a0 00 00 34 ┊ 60 5e 40 b9 fd 7b 41 a9 │h×_××004┊`^@××{A×│
│000d53c0│ f4 4f c2 a8 dc d7 ff 17 ┊ fd 7b 41 a9 f4 4f c2 a8 │×O×××××•┊×{A××O××│
│000d53d0│ c0 03 5f d6 ff c3 00 d1 ┊ fd 7b 02 a9 fd 83 00 91 │×•_×××0×┊×{•×××0×│
│000d53e0│ 9e 59 00 94 84 00 00 b4 ┊ e3 03 1f aa fd 7b 42 a9 │×Y0××00×┊×••××{B×│
│000d53f0│ 4d 4f 00 14 2a 43 28 30 ┊ 69 42 28 10 40 41 28 70 │MO0•*C(0┊iB(•@A(p│
│000d5400│ 1f 20 03 d5 68 28 80 52 ┊ 1f 20 03 d5 1f 20 03 d5 │• •×h(×R┊• •×• •×│
│000d5410│ 9d 4e 00 94 ff c3 00 d1 ┊ fd 7b 02 a9 fd 83 00 91 │×N0×××0×┊×{•×××0×│
│000d5420│ 88 00 40 f9 1f 01 02 eb ┊ 62 00 00 54 b6 67 00 94 │×0@×•••×┊b00T×g0×│
│000d5430│ 0f 00 00 14 e8 03 01 2a ┊ e9 03 1f aa 3f 01 02 eb │•00•×••*┊×••×?••×│
│000d5440│ 22 01 00 54 0a 60 40 39 ┊ 8a 01 00 36 0a 04 40 f9 │"•0T_`@9┊×•06_•@×│
│000d5450│ 0b 01 09 8b 6a 69 6a b8 ┊ 6a 68 29 b8 29 11 00 91 │••_×jij×┊jh)×)•0×│
│000d5460│ f7 ff ff 17 e0 03 1f 2a ┊ 82 00 00 f9 fd 7b 42 a9 │×××•×••*┊×00××{B×│
│000d5470│ ff c3 00 91 c0 03 5f d6 ┊ ca db 27 10 c9 c3 27 70 │××0××•_×┊××'•××'p│
│000d5480│ 20 3d 28 70 1f 20 03 d5 ┊ c8 5b 80 52 1f 20 03 d5 │ =(p• •×┊×[×R• •×│
│000d5490│ 1f 20 03 d5 7c 4e 00 94 ┊ ff 83 02 d1 f4 4f 08 a9 │• •×|N0×┊××•××O•×│
│000d54a0│ fd 7b 09 a9 fd 43 02 91 ┊ f3 03 01 2a e1 03 02 aa │×{_××C•×┊×••*×••×│
│000d54b0│ 30 5d 00 94 d0 6a 00 94 ┊ f4 60 00 94 81 02 80 52 │0]0××j0×┊×`0××•×R│
│000d54c0│ f3 0b 00 b9 c4 ff ff 97 ┊ fd 7b 49 a9 95 60 00 14 │×•0×××××┊×{I××`0•│
```

Static Reverse Engineering 🐉


Binary
Decompiled
Disassembled
CodeBrowser: hwioctf:/a.out
File Edit Analysis Graph Navigation Search Select Tools Window Help
Bytes: a.out
Listing: a.out
Decompile: main - (a....

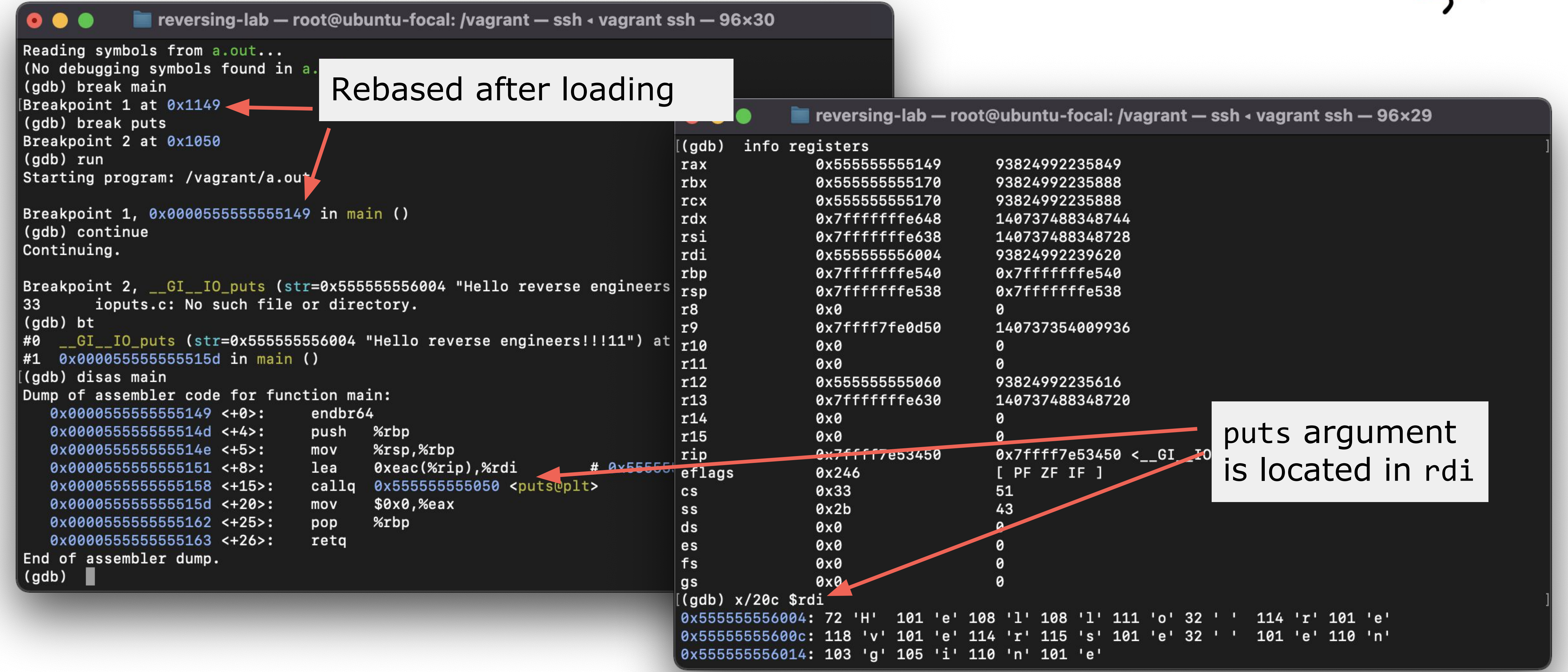
```
Addresses   Hex
00101148    ff f3 0f 1e fa 55 48 89
00101150    e5 48 8d 3d ac 0e 00 00
00101158    e8 f3 fe ff ff b8 00 00
00101160    00 00 5d c3 66 2e 0f 1f
00101168    84 00 00 00 00 00 66 90
00101170    f3 0f 1e fa 41 57 4c 8d
00101178    3d 3b 2c 00 00 41 56 49
00101180    89 d6 41 55 49 89 f5 41
00101188    54 41 89 fc 55 48 8d 2d
00101190    2c 2c 00 00 53 4c 29 fd
00101198    48 83 ec 08 e8 5f fe ff
001011a0    ff 48 c1 fd 03 74 1f 31
001011a8    db 0f 1f 80 00 00 00 00
001011b0    4c 89 f2 4c 89 ee 44 89
001011b8    e7 41 ff 14 df 48 83 c3
001011c0    01 48 39 dd 75 ea 48 83
001011c8    c4 08 5b 5d 41 5c 41 5d
001011d0    41 5e 41 5f c3 66 66 2e
001011d8    0f 1f 84 00 00 00 00 00
001011e0    f3 0f 1e fa c3

Start: 00100000 End: _elfSectionHeaders::...Offset: 00000000 Insertion: 00101158
```

```
                    ****************************************************************
                    *                          FUNCTION                            *
                    ****************************************************************
                    undefined main()
     undefined         AL:1           <RETURN>
                    main                                     XREF[4]:    Entry Point
                                                                         _start:0010
                                                                         001020f8(*)
00101149 f3 0f 1e fa     ENDBR64
0010114d 55              PUSH        RBP
0010114e 48 89 e5        MOV         RBP,RSP
00101151 48 8d 3d        LEA         RDI,[s_Hello_reverse_engineers!!!11_00102004]     = "Hello
         ac 0e 00 00
00101158 e8 f3 fe        CALL        <EXTERNAL>::puts                                  int puts
         ff ff
0010115d b8 00 00        MOV         EAX,0x0
         00 00
00101162 5d              POP         RBP
00101163 c3              RET
00101164 66              ??          66h    f
00101165 2e              ??          2Eh    .
```

```
undefined8 main(void)

{
  puts("Hello reverse engineer
  return 0;
}
```

```
00101158    main    CALL 0x00101050
```

Dynamic Analysis (e.g. `gdb`)


reversing-lab — root@ubuntu-focal: /vagrant — ssh ◂ vagrant ssh — 96×30
Rebased after loading
reversing-lab — root@ubuntu-focal: /vagrant — ssh ◂ vagrant ssh — 96×29
puts argument is located in rdi

```
Reading symbols from a.out...
(No debugging symbols found in a.
(gdb) break main
Breakpoint 1 at 0x1149
(gdb) break puts
Breakpoint 2 at 0x1050
(gdb) run
Starting program: /vagrant/a.out

Breakpoint 1, 0x0000555555555149 in main ()
(gdb) continue
Continuing.

Breakpoint 2, __GI__IO_puts (str=0x555555556004 "Hello reverse engineers
33      ioputs.c: No such file or directory.
(gdb) bt
#0  __GI__IO_puts (str=0x555555556004 "Hello reverse engineers!!!11") at
#1  0x000055555555515d in main ()
(gdb) disas main
Dump of assembler code for function main:
   0x0000555555555149 <+0>:     endbr64
   0x000055555555514d <+4>:     push   %rbp
   0x000055555555514e <+5>:     mov    %rsp,%rbp
   0x0000555555555151 <+8>:     lea    0xeac(%rip),%rdi        # 0x55555
   0x0000555555555158 <+15>:    callq  0x555555555050 <puts@plt>
   0x000055555555515d <+20>:    mov    $0x0,%eax
   0x0000555555555162 <+25>:    pop    %rbp
   0x0000555555555163 <+26>:    retq
End of assembler dump.
(gdb)
```

```
(gdb)  info registers
rax            0x555555555149      93824992235849
rbx            0x555555555170      93824992235888
rcx            0x555555555170      93824992235888
rdx            0x7fffffffe648      140737488348744
rsi            0x7fffffffe638      140737488348728
rdi            0x555555556004      93824992239620
rbp            0x7fffffffe540      0x7fffffffe540
rsp            0x7fffffffe538      0x7fffffffe538
r8             0x0                 0
r9             0x7ffff7fe0d50      140737354009936
r10            0x0                 0
r11            0x0                 0
r12            0x555555555060      93824992235616
r13            0x7fffffffe630      140737488348720
r14            0x0                 0
r15            0x0                 0
rip            0x7ffff7e53450      0x7ffff7e53450 <__GI__IO
eflags         0x246               [ PF ZF IF ]
cs             0x33                51
ss             0x2b                43
ds             0x0                 0
es             0x0                 0
fs             0x0                 0
gs             0x0                 0
(gdb) x/20c $rdi
0x555555556004: 72 'H'  101 'e' 108 'l' 108 'l' 111 'o' 32 ' '  114 'r' 101 'e'
0x55555555600c: 118 'v' 101 'e' 114 'r' 115 's' 101 'e' 32 ' '  101 'e' 110 'n'
0x555555556014: 103 'g' 105 'i' 110 'n' 101 'e'
```

🐢 Static vs. Dynamic 🐇

- Static 🐢
  - Everything is in the binary!
  - Extensive staring will bring back (almost) the original source code.

- Dynamic 🐇
  - Execute the binary and wait what happens.
  - Very fast if you know what you’re looking for.
  - Might miss details and certain conditions that first have to be found statically.

You’ll need different tools for both analysis methods.

Hooking

The heavy lifting of reverse engineering.

Hooking

Print `beneficiary->name` and `credit`

Change `beneficiary` or `credit`

```c
int handle_transfer(struct account* beneficiary, int credit) {

   // nothing was transferred, keep the money
   if !account_exists(beneficiary)
       return 0;

   // transfer and confirm money
   beneficiary->value += credit;
   return credit;

}
```

Return `0` to pretend we didn’t get the money 🤑

Hooking Methods

Static 🐢

Patch the binary, then run it.

Dynamic 🐇

Run the binary, then patch it.

Keep most of the program intact, only hook very specific parts.

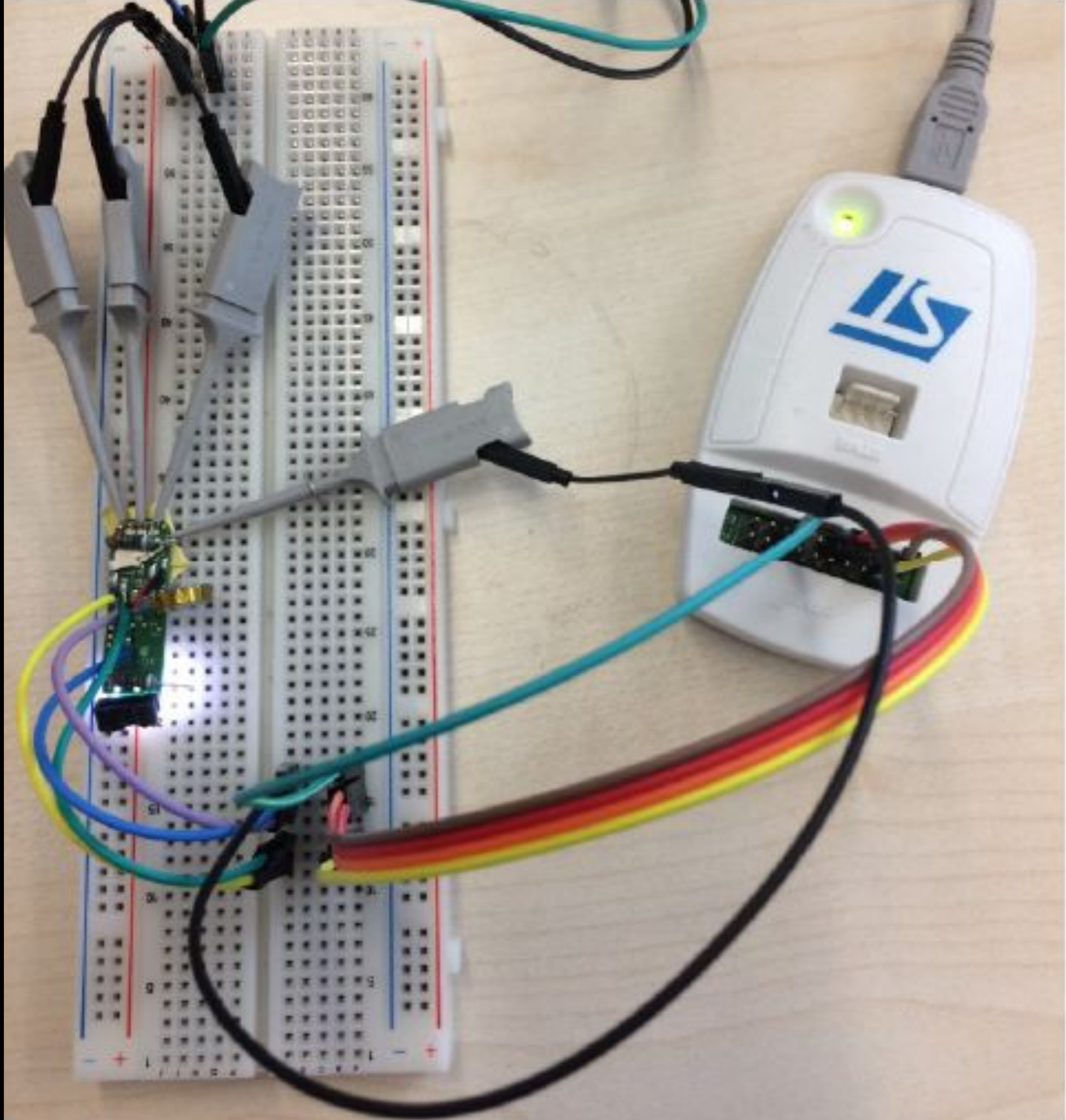
Firmware Reversing

“What you’re doing is really challenging.”

🏁 Play CTFs if you would like to know a difficulty level or get a solution later on.

You wouldn’t simply attach gdb
… or would you?

Running gdb on the Fitbit
requires patched firmware.
🥚🐔



Code? Data?

Your disassembler might get 25% of function starts wrong on raw Arm firmware.

SoK: All You Ever Wanted to Know About x86/x64 Binary Disassembly But Were Afraid to Ask

Chengbin Pang*‡§ Ruotong Yu* Yaohui Chen† Eric Koskinen* Georgios Portokalidis* Bing Mao‡ Jun Xu*

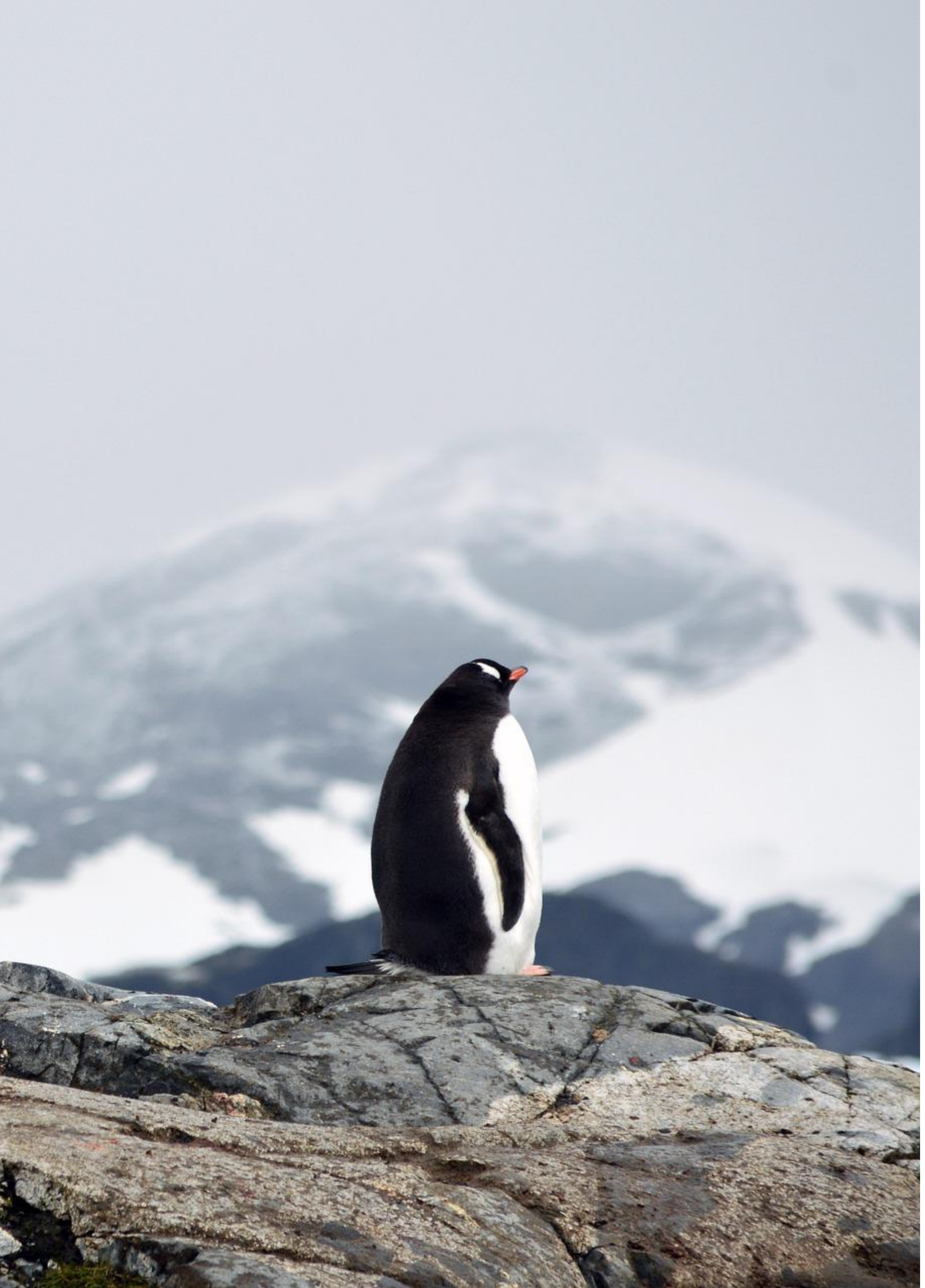
*Stevens Institute of Technology †Facebook Inc. ‡Nanjing University

***Abstract*—Disassembly of binary code is hard, but necessary for improving the security of binary software. Over the past few decades, research in binary disassembly has produced many tools and frameworks, which have been made available to researchers and security professionals. These tools employ a variety of strategies that grant them different**

The lack of

TABLE I: The group of open-source tools that our study covers and representative works that use those tools.

Tool (Version)	Source (Release Date)
PSI (1.0)	Website [62]



Reversing Open-Source

- What you’re reversing might be open-source.
- Look for specific libraries, e.g., encryption, real-time operating systems, …
- Get different firmware versions for your target!

```
00081560  32 32 32 00 4c e4 05 00  3e 24 00 00 01 08 6e 04  |222.L...>$....n.|
00081570  02 00 00 00 34 4e 56 60  78 e4 05 00 02 20 00 00  |....4NV`x.... ..|
00081580  1b 00 20 00 28 00 0f 0f  0f 00 00 00 6c e5 05 00  |.. .(.......l...|
00081590  00 00 00 00 43 6f 70 79  72 69 67 68 74 20 28 63  |....Copyright (c|
000815a0  29 20 31 39 39 36 2d 32  30 30 33 20 45 78 70 72  |) 1996-2003 Expr|
000815b0  65                                            2a  |ess Logic Inc. *|
000815c0  20                                          f 47  | ThreadX ATMEL/G|
000815d0  72                                            69  |reen Hills Versi|
000815e0  6f 6e 20 47 34 2e 30 62  2e 34 2e 30 63 20 2a 00  |on G4.0b.4.0c *.|
000815f0  47 2d 47 42 2d 47 4c 2d  4d 2d 44 2d 44 4c 2d 4b  |G-GB-GL-M-D-DL-K|
00081600  4d 4c 2d 43 4d 52 2d 48  4d 52 2d 4d 4c 32 2d 47  |ML-CMR-HMR-ML2-G|
00081610  5a 2d 4b 48 32 2d 43 4d  2d 52 50 2d 54 43 2d 4e  |Z-KH2-CM-RP-TC-N|
00081620  48 2d 54 44 2d 41 50 2d  48 41 2d 47 46 2d 44 44  |H-TD-AP-HA-GF-DD|
00081630  2d 41 54 2d 4d 46 2d 4d  53 2d 44 57 2d 55 53 41  |-AT-MF-MS-DW-USA|
00081640  2d 43 41 2d 53 44 2d 53  44 53 55 00 05 00 00 00  |-CA-SD-SDSU.....|
00081650  08 00 0f 00 1a 00 00 00  28 46 6c 6f 61 74 69 6e  |........(Floatin|
00081660  67 20 70 6f 69 6e 74 20  6f 75 74 70 75 74 20 75  |g point output u|
00081670  6e 73 75 70 70 6f 72 74  65 64 20 77 2f 2d 6e 6f  |nsupported w/-no|
```

There's ThreadX documentation and newer versions are open-source!

From the Bluetooth firmware that Dennis and me were staring on for multiple months, older BCM20702 version.

```
002069d0  00 00 00 00 55 45 55 51  00 00 00 00 02 00 00 00  |....UEUQ........|
002069e0  01 00 00 00 00 00 00 00  01 00 00 00 10 0d 20 00  |.............. .|
002069                                                         ... ... .....|
00206a                                                         ..j .li .UEUQ|
00206a                                                         .............|
00206a                                                         ... .  . ... .|
00206a                                                      |.. .........Dj .|
00206a40  d4 69 20 00 55 45 55 51  00 00 00 00 02 00 00 00  |.i .UEUQ........|
00206a50  20 00 00 00 00 00 00 00  20 00 00 00 7c 6a 20 00  | ....... ...|j .|
00206a60  7c 6b 20 00 7c 6a 20 00  7c 6a 20 00 00 00 00 00  ||k .|j .|j .....|
00206a70  00 00 00 00 90 33 20 00  0c 6a 20 00 00 00 00 00  |.....3 ..j .....|
00206a80  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00206b70  00 00 00 00 00 00 00 00  00 00 00 00 4e 44 56 44  |............NDVD|
00206b80  00 00 00 00 00 00 00 00  00 00 00 00 c8 6b 20 00  |.............k .|
00206b90  01 00 00 00 ac 9c 20 00  44 35 20 00 00 00 00 00  |...... .D5 .....|
00206ba0  4c 36 20 00 7f a7 00 00  00 00 00 00 00 00 00 00  |L6 .............|
00206bb0  00 00 00 00 45 a7 00 00  00 00 00 00 00 00 00 00  |....E...........|
00206bc0  00 00 00 00 00 00 00 00  44 52 48 54 6b 02 00 00  |........DRHTk...|
00206bd0  80 6d 20 00 6c 6c 20 00  47 6e 20 00 dc 01 00 00  |.m .ll .Gn .....|
```

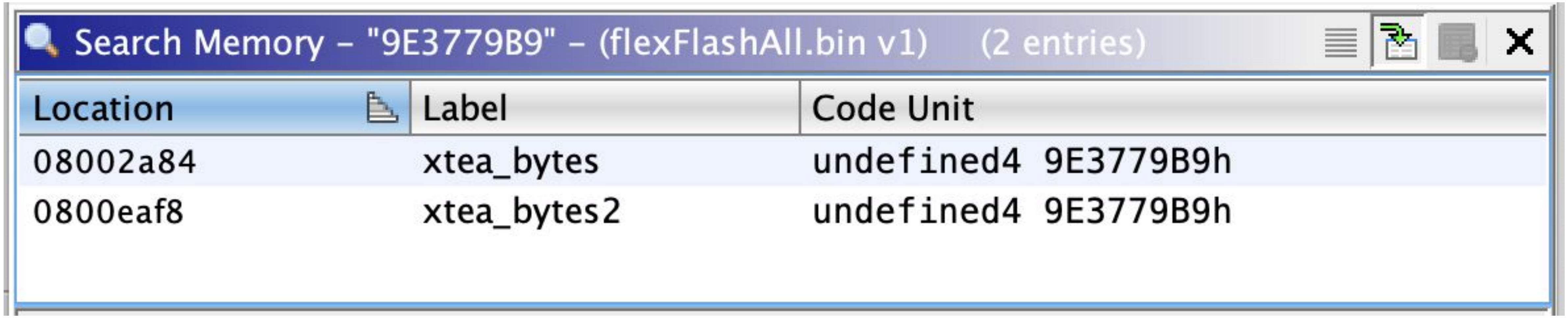
```
/* Define queue control specific data definitions.  */

#define TX_QUEUE_ID         ((ULONG) 0x51554555)
```

https://github.com/azure-rtos/threadx/blob/master/common/inc/tx_queue.h

Even if you don’t find source code for an RTOS,
reversing semantics of threads, queues, etc.
will be a great starting point!

Encryption

- Cryptographic algorithms use magic numbers.
- XTEA encryption delta value: `0x9E3779B9`
- …how many encryption libraries supporting 32bit Arm are there?

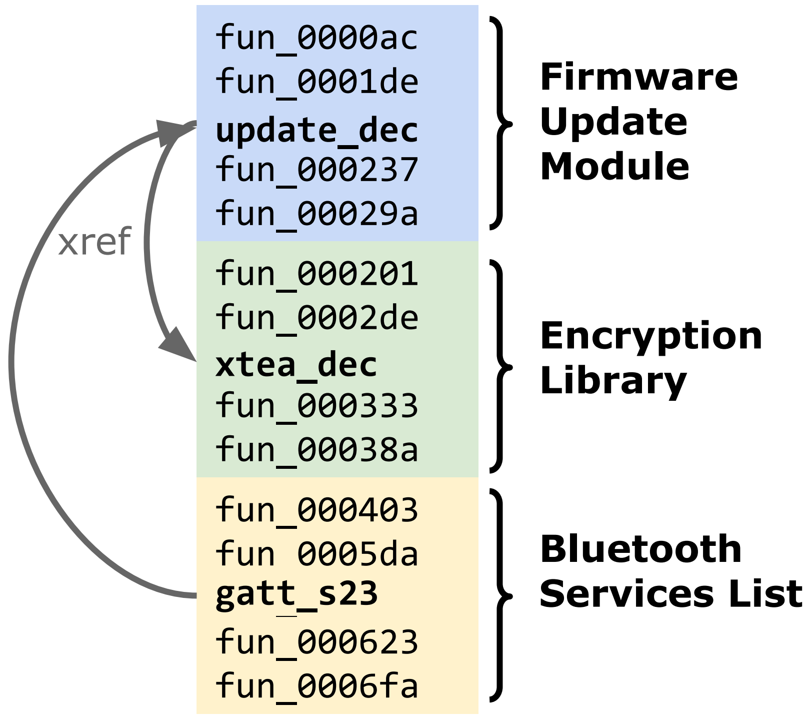


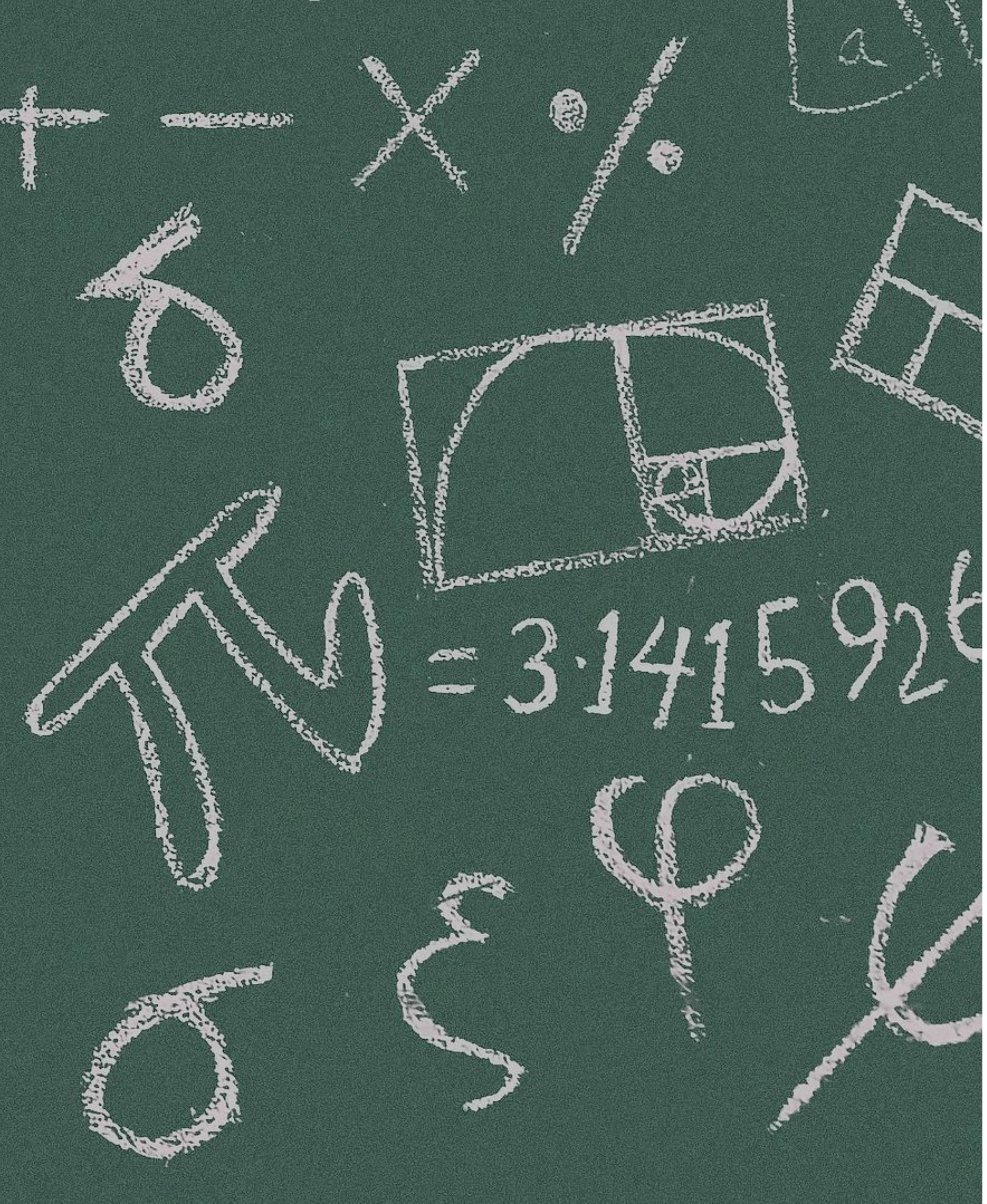
Search Memory - "9E3779B9" - (flexFlashAll.bin v1) (2 entries)

Location	Label	Code Unit
08002a84	xtea_bytes	undefined4 9E3779B9h
0800eaf8	xtea_bytes2	undefined4 9E3779B9h

Statically-Linked Libraries

- If code from multiple libraries/modules is included in the same binary, the compiler tends to keep them in the same location.
- Some libraries/algorithms/specifications are open-source, search for weird numbers online.
- Check if nearby functions and xrefs belong to the same library. Note that cross-references are not always found by the disassembler!



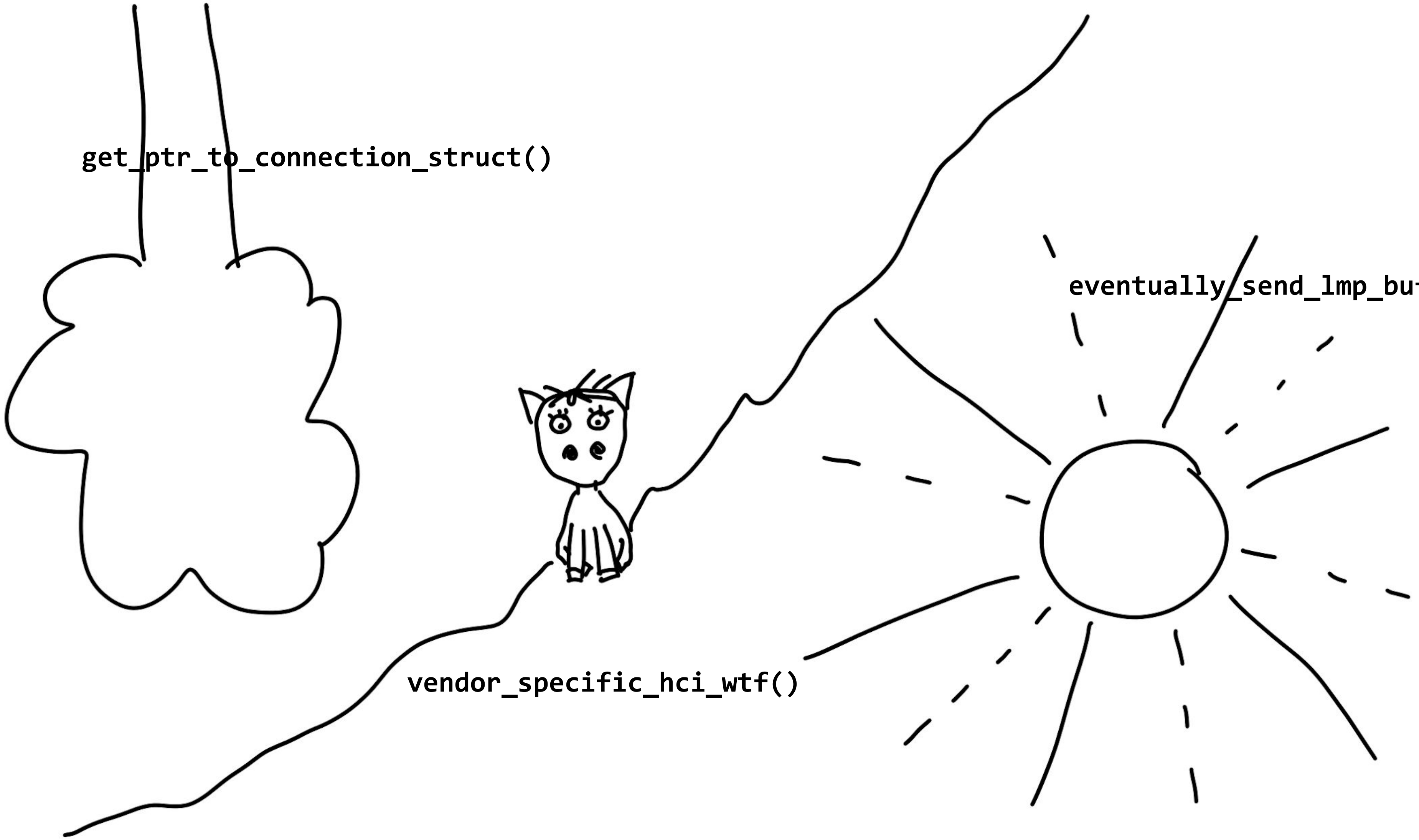


Symbols

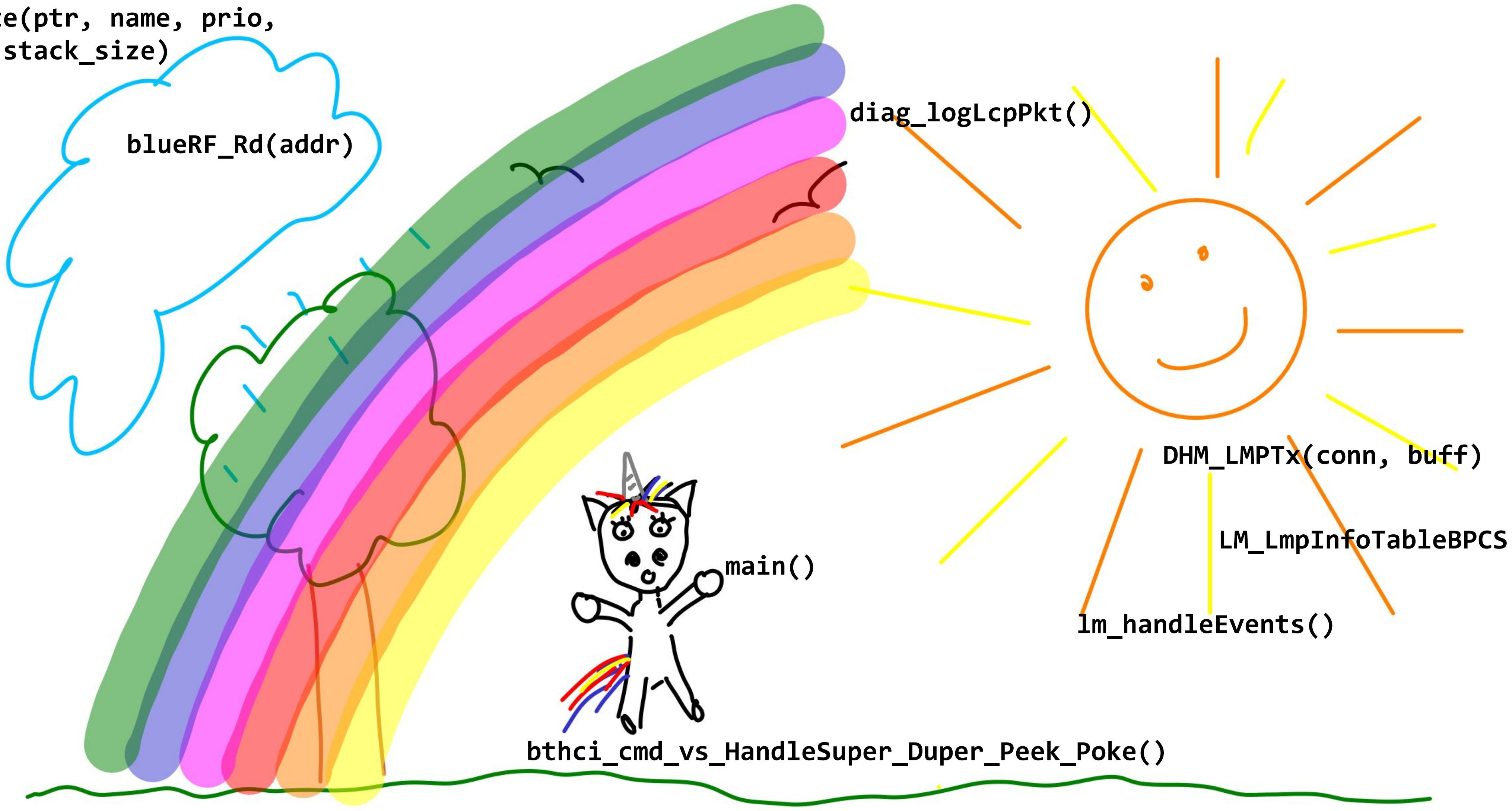
- Raw firmware flash dumps don’t contain symbols.
- SDKs might still contain symbols to link components into the firmware!

Reviewer 2:
“Does it work on the latest firmware?”

Reverse Engineering without Symbols


get_ptr_to_connection_struct()
eventually_send_lmp_buffer()
vendor_specific_hci_wtf()

Reverse Engineering with Symbols


thread_Create(ptr, name, prio, func, 0, 0, stack_size)
blueRF_Rd(addr)
diag_logLcpPkt()
DHM_LMPTx(conn, buff)
LM_LmpInfoTableBPCS
lm_handleEvents()
main()
bthci_cmd_vs_HandleSuper_Duper_Peek_Poke()

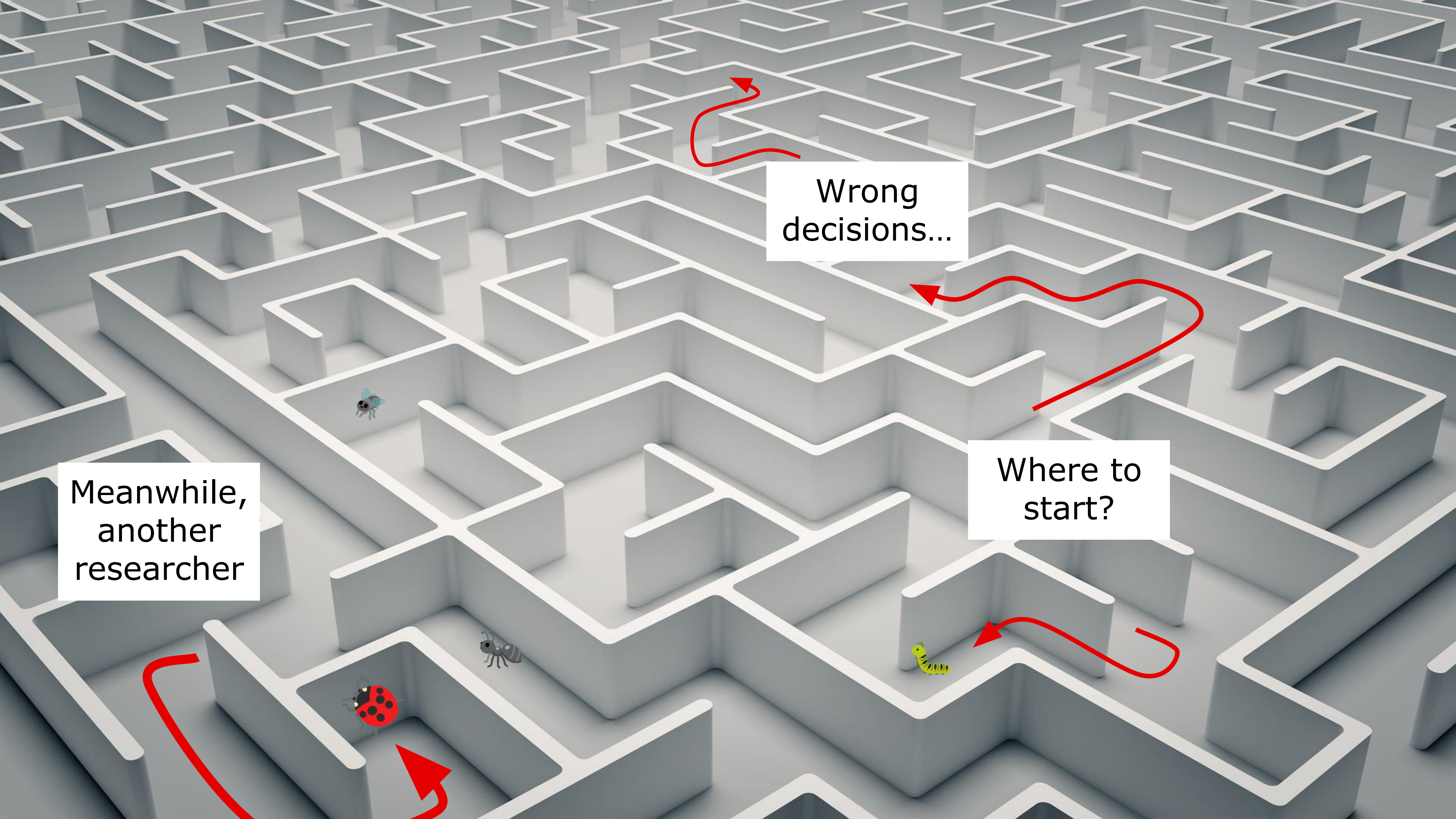
Real-World Targets

Specify a Goal

- Can we send custom data or waveforms with this Wi-Fi chip?
- Could this protocol be more performant when we change the scheduling?
- How does Apple’s Bluetooth stack behave when we pick a high Ratchet during the MagicPairing algorithm?

Anything that keeps <u>you</u> motivated!

- What you’re looking for might not exist.
- Real-world targets are overwhelmingly large.
- Most bugs are relatively simple (compared to advanced CTFs).
- Certain attack vectors might’ve been extensively tested by someone else before.


Wrong decisions…
Where to start?
Meanwhile, another researcher

Getting Started

Great Resources

- begin.re
  Get started with Windows/x86 reverse engineering and hack Minesweeper.

- ragingrock.com/AndroidAppRE
  Learn how to reverse engineer Android apps (Java/x64) to uncover malware functionality.

Don’t give up staring!

Q&A

youtube.com/@jiskac

@jiska@chaos.social

github.com/seemoo-lab

jclassen@seemoo.de