Light side of reverse engineering

...

Boris Ryutin
@dukebarman

whoami

- Security Engineers Lead at 🅨
- Dev**Sec**Ops
- Speaker
- Author of @dukebarmanpro & @fuzzing_life
- ❤️ OpenSource
  - https://github.com/bondifuzz
  - ...

Agenda

1. Intro
2. Types
3. Ways
4. Examples
5. Tools
6. Demo
7. Afterword

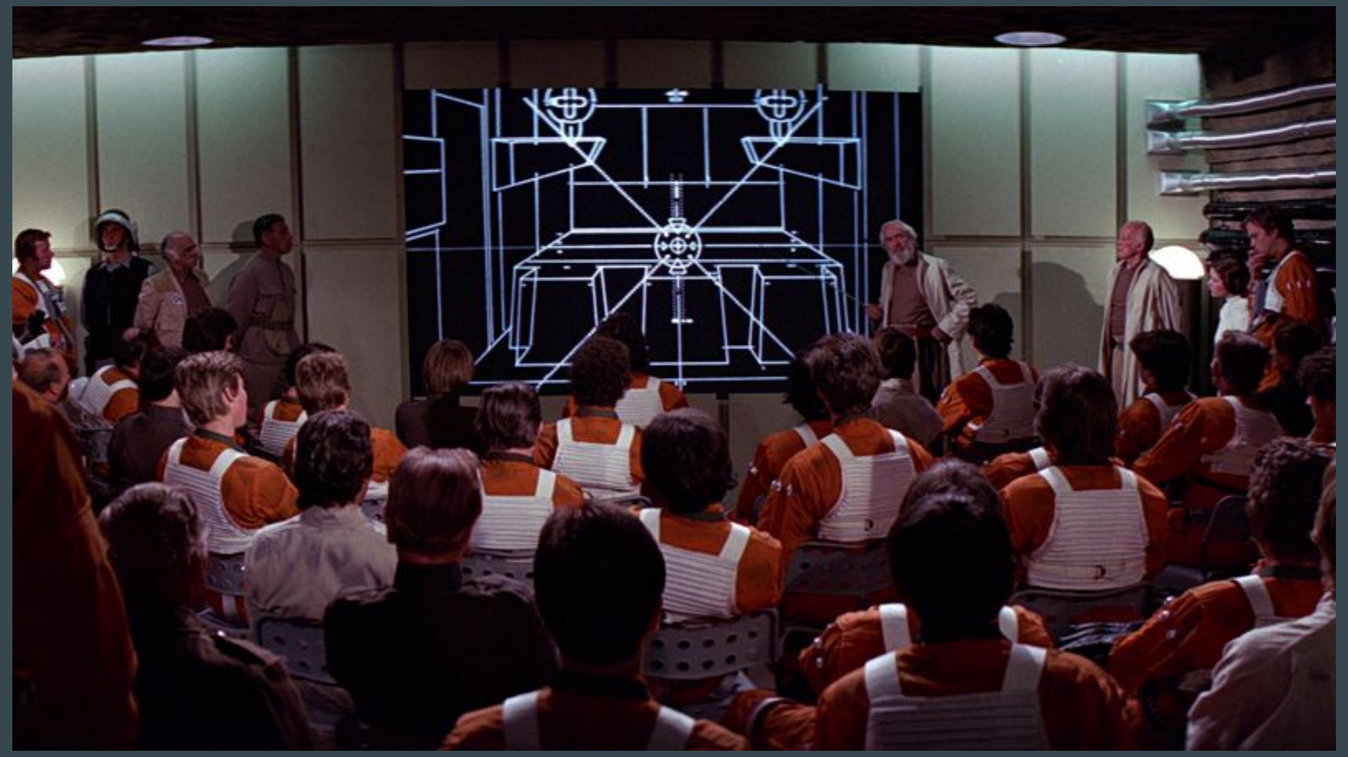


Image by Lucasfilm Ltd

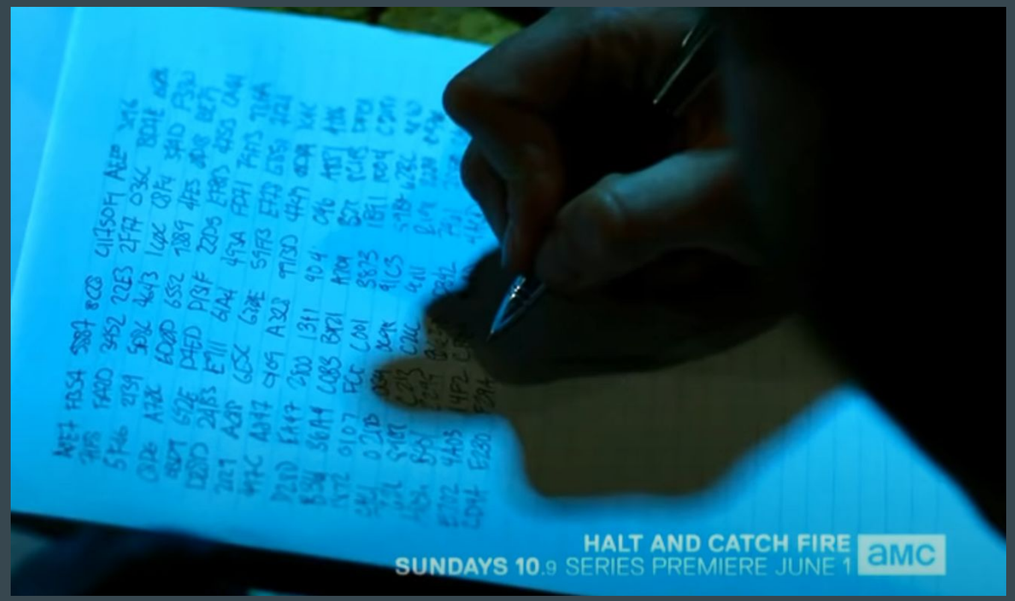
Business analyst vs. Reverse analyst

“Reverse engineering or backward engineering is the process of deconstructing a product to see how it works”

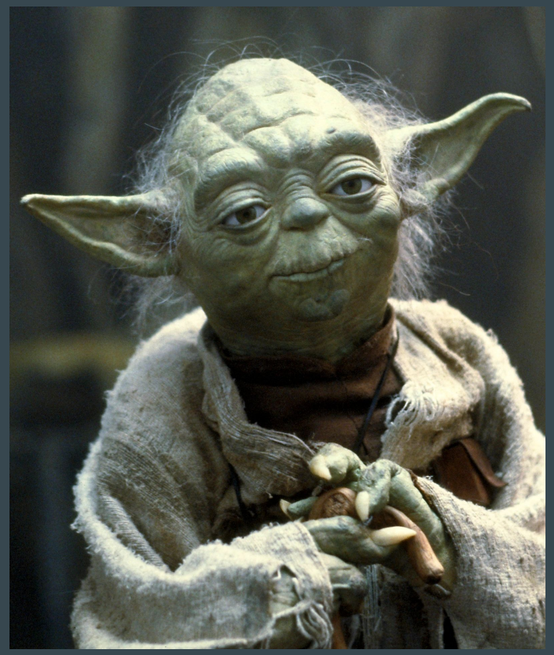
Image by Lucasfilm Ltd

What is Reverse Engineering?

- BIOS (Halt and Catch fire)
- Samba (Linux...)
- ...



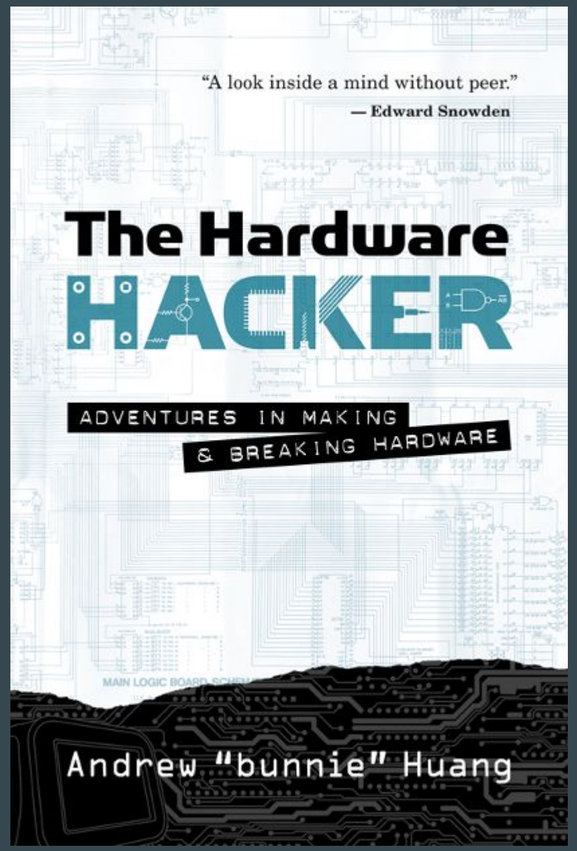
Types



How many?

Image by Lucasfilm Ltd

Types


"A look inside a mind without peer."
— Edward Snowden
The Hardware
HACKER
ADVENTURES IN MAKING
& BREAKING HARDWARE
Andrew "bunnie" Huang

Examples

Examples. Legacy

Only legacy without source code is worse than a legacy

Image by Lucasfilm Ltd

Examples. Porting

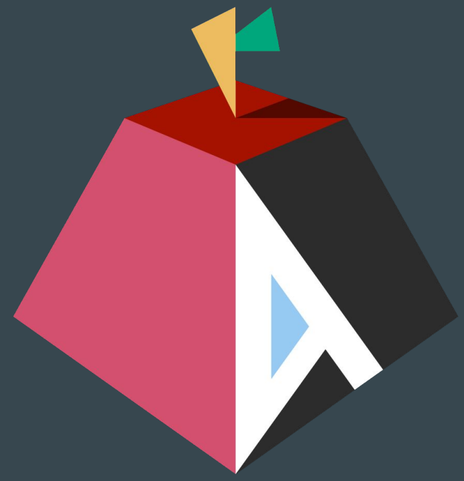
- ReactOS
- Wine
- Apple Game Porting Toolkit
- Asahi Linux

The Register

This article is more than **1 year old**

Linus Torvalds releases Linux 5.19 – using Asahi on an Arm-powered Mac

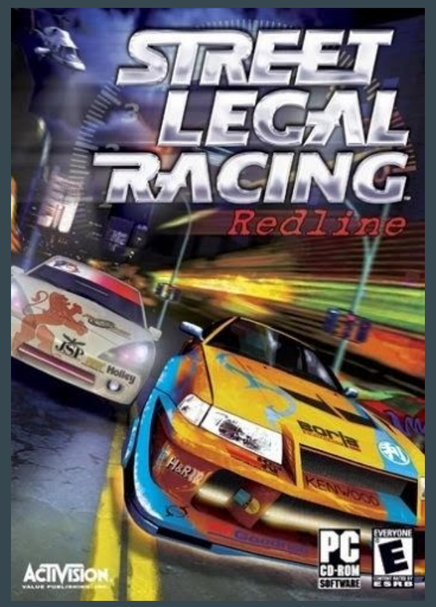
Hails the combo as finally making Arm 'usable as a development platform'

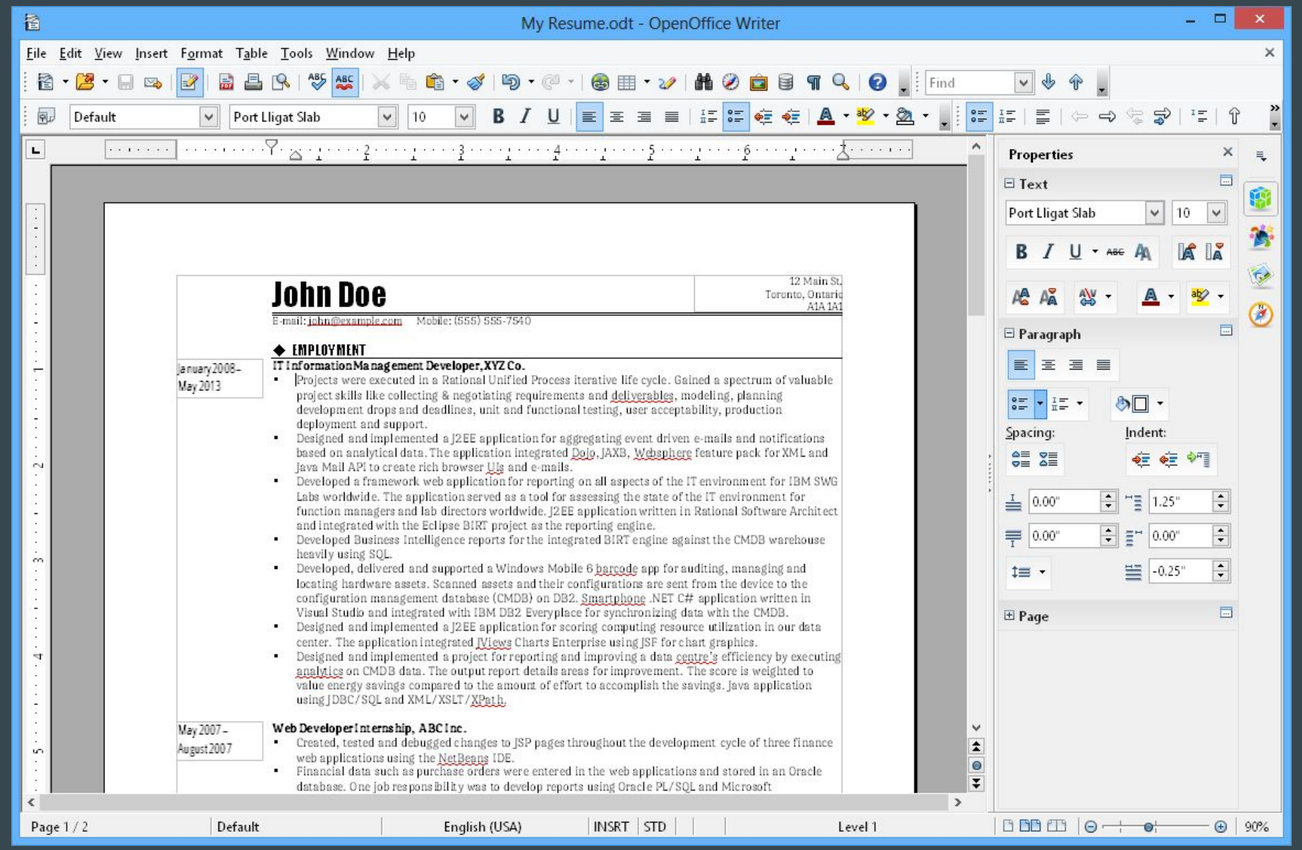


Asahi Linux

Examples. New functionality

- OpenOffice (DOC)
- Graphics editors
- Street Legal Racing (multiplayer)

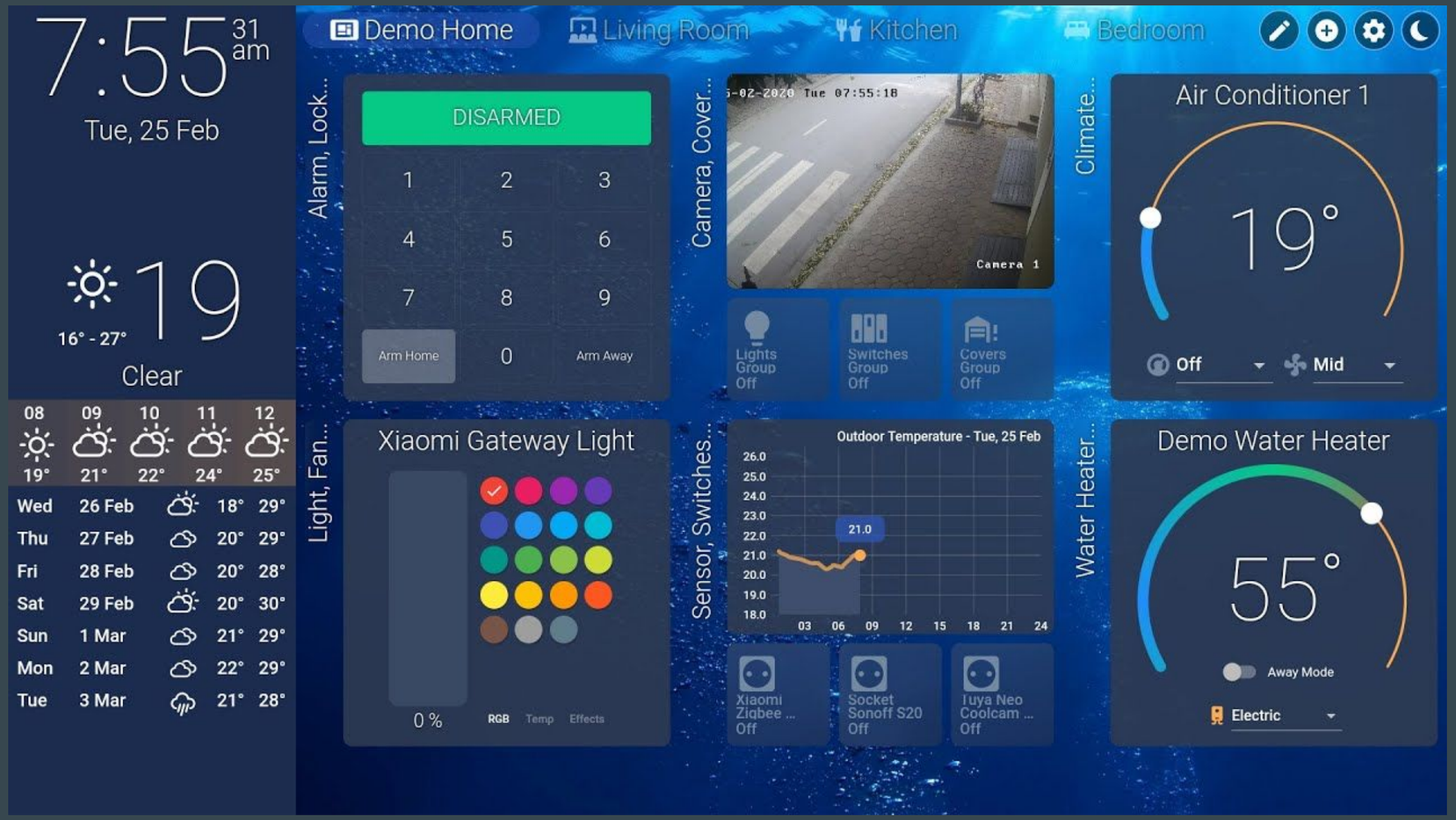




Examples. Heroes 3 - HotA

Examples. Hardware integration

- Home Assistant
- Old devices



Examples. Recovery data

- manual
- automate

Image by Lucasfilm Ltd

Examples. InfoSec

- Malware Analysis
- Security Audit

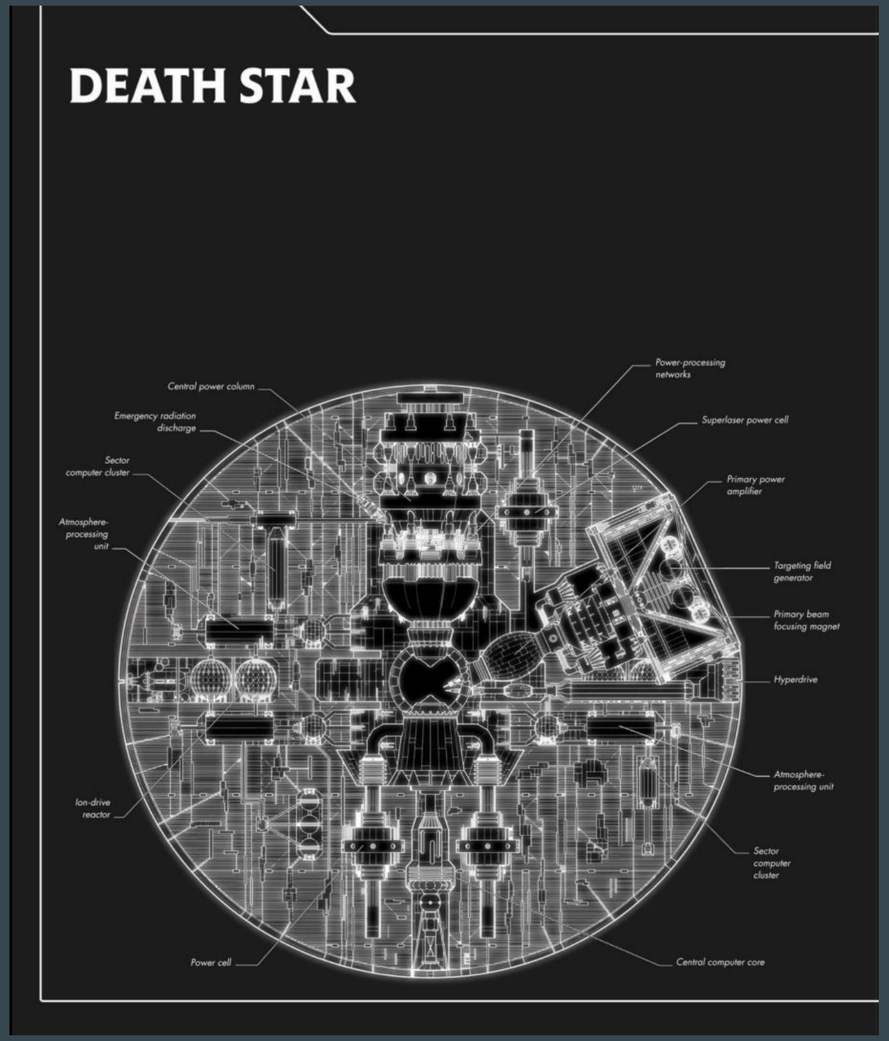


Image by Lucasfilm Ltd

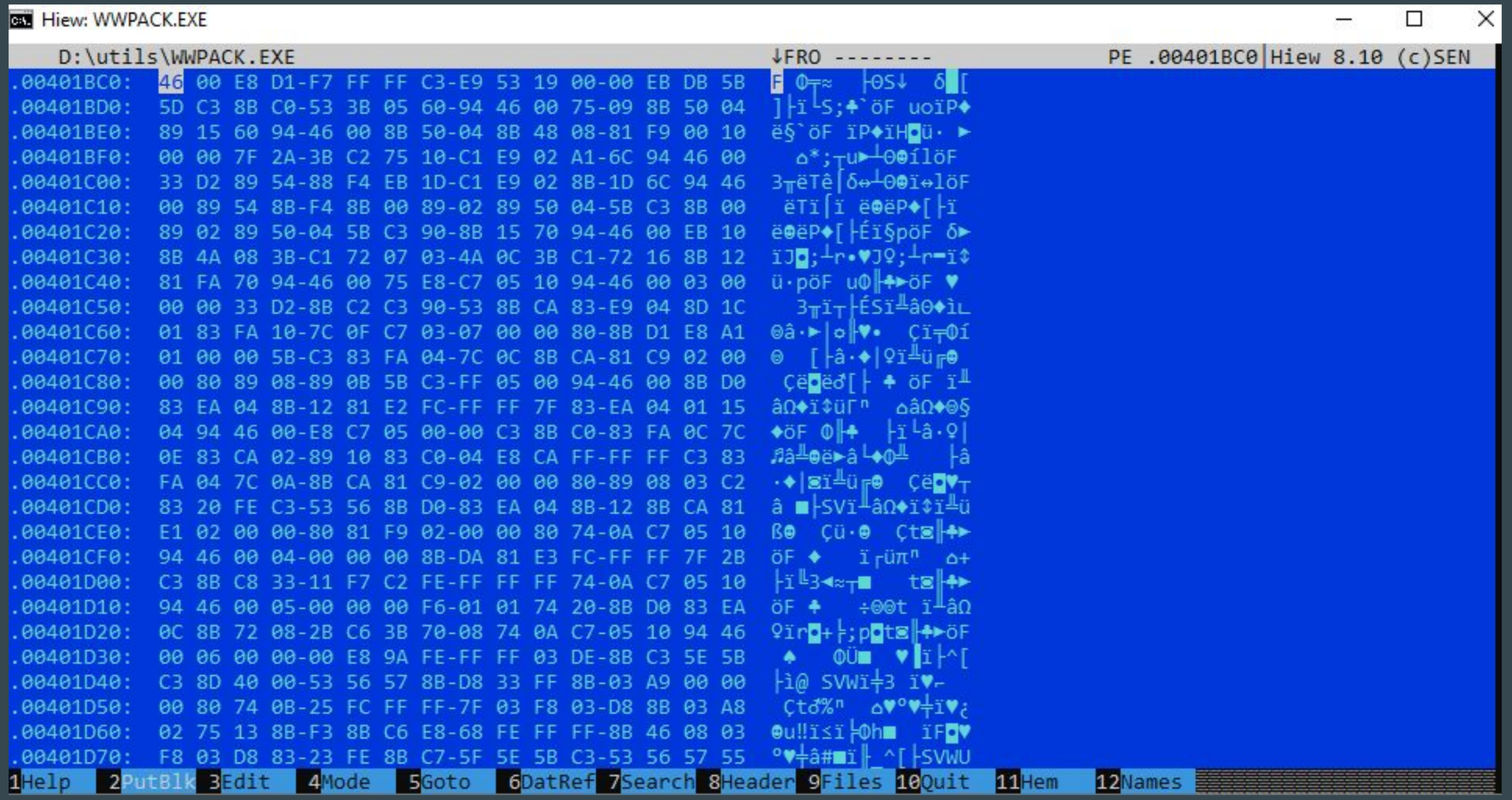
Tools

Tools. Static or Dynamic side?

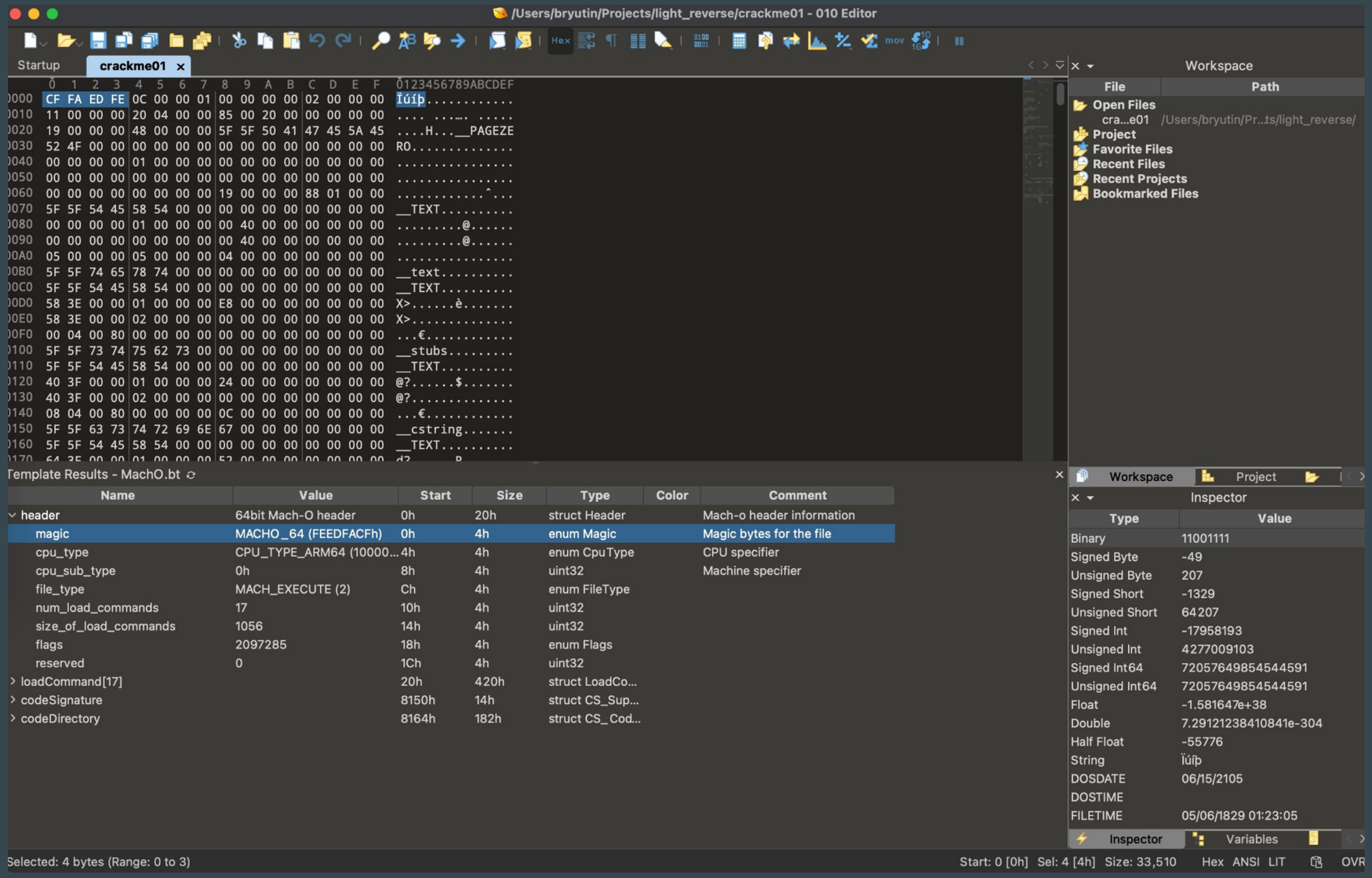
- hex editors
- disassemblers + decompilers
- debuggers
- DBI
- ...?

Binary... 3 steps of diving

HIEW


Hiew: WWPACK.EXE
D:\utils\WWPACK.EXE
↓FRO --------
PE .00401BC0 Hiew 8.10 (c)SEN
.00401BC0: 46 00 E8 D1-F7 FF FF C3-E9 53 19 00-00 EB DB 5B
.00401BD0: 5D C3 8B C0-53 3B 05 60-94 46 00 75-09 8B 50 04
.00401BE0: 89 15 60 94-46 00 8B 50-04 8B 48 08-81 F9 00 10
.00401BF0: 00 00 7F 2A-3B C2 75 10-C1 E9 02 A1-6C 94 46 00
.00401C00: 33 D2 89 54-88 F4 EB 1D-C1 E9 02 8B-1D 6C 94 46
.00401C10: 00 89 54 8B-F4 8B 00 89-02 89 50 04-5B C3 8B 00
.00401C20: 89 02 89 50-04 5B C3 90-8B 15 70 94-46 00 EB 10
.00401C30: 8B 4A 08 3B-C1 72 07 03-4A 0C 3B C1-72 16 8B 12
.00401C40: 81 FA 70 94-46 00 75 E8-C7 05 10 94-46 00 03 00
.00401C50: 00 00 33 D2-8B C2 C3 90-53 8B CA 83-E9 04 8D 1C
.00401C60: 01 83 FA 10-7C 0F C7 03-07 00 00 80-8B D1 E8 A1
.00401C70: 01 00 00 5B-C3 83 FA 04-7C 0C 8B CA-81 C9 02 00
.00401C80: 00 80 89 08-89 0B 5B C3-FF 05 00 94-46 00 8B D0
.00401C90: 83 EA 04 8B-12 81 E2 FC-FF FF 7F 83-EA 04 01 15
.00401CA0: 04 94 46 00-E8 C7 05 00-00 C3 8B C0-83 FA 0C 7C
.00401CB0: 0E 83 CA 02-89 10 83 C0-04 E8 CA FF-FF FF C3 83
.00401CC0: FA 04 7C 0A-8B CA 81 C9-02 00 00 80-89 08 03 C2
.00401CD0: 83 20 FE C3-53 56 8B D0-83 EA 04 8B-12 8B CA 81
.00401CE0: E1 02 00 00-80 81 F9 02-00 00 80 74-0A C7 05 10
.00401CF0: 94 46 00 04-00 00 00 8B-DA 81 E3 FC-FF FF 7F 2B
.00401D00: C3 8B C8 33-11 F7 C2 FE-FF FF FF 74-0A C7 05 10
.00401D10: 94 46 00 05-00 00 00 F6-01 01 74 20-8B D0 83 EA
.00401D20: 0C 8B 72 08-2B C6 3B 70-08 74 0A C7-05 10 94 46
.00401D30: 00 06 00 00-00 E8 9A FE-FF FF 03 DE-8B C3 5E 5B
.00401D40: C3 8D 40 00-53 56 57 8B-D8 33 FF 8B-03 A9 00 00
.00401D50: 00 80 74 0B-25 FC FF FF-7F 03 F8 03-D8 8B 03 A8
.00401D60: 02 75 13 8B-F3 8B C6 E8-68 FE FF FF-8B 46 08 03
.00401D70: F8 03 D8 83-23 FE 8B C7-5F 5E 5B C3-53 56 57 55
1Help 2PutBlk 3Edit 4Mode 5Goto 6DatRef 7Search 8Header 9Files 10Quit 11Hem 12Names

010 Editor


/Users/bryutin/Projects/light_reverse/crackme01 - 010 Editor
Startup
crackme01
Workspace
File
Path
Open Files
cra...e01 /Users/bryutin/Pr...ts/light_reverse/
Project
Favorite Files
Recent Files
Recent Projects
Bookmarked Files
Template Results - MachO.bt
Name
Value
Start
Size
Type
Color
Comment
header
64bit Mach-O header
0h
20h
struct Header
Mach-o header information
magic
MACHO_64 (FEEDFACFh)
0h
4h
enum Magic
Magic bytes for the file
cpu_type
CPU_TYPE_ARM64 (10000...
4h
4h
enum CpuType
CPU specifier
cpu_sub_type
0h
8h
4h
uint32
Machine specifier
file_type
MACH_EXECUTE (2)
Ch
4h
enum FileType
num_load_commands
17
10h
4h
uint32
size_of_load_commands
1056
14h
4h
uint32
flags
2097285
18h
4h
enum Flags
reserved
0
1Ch
4h
uint32
loadCommand[17]
20h
420h
struct LoadCo...
codeSignature
8150h
14h
struct CS_Sup...
codeDirectory
8164h
182h
struct CS_Cod...
Workspace
Project
Inspector
Type
Value
Binary
11001111
Signed Byte
-49
Unsigned Byte
207
Signed Short
-1329
Unsigned Short
64207
Signed Int
-17958193
Unsigned Int
4277009103
Signed Int64
72057649854544591
Unsigned Int64
72057649854544591
Float
-1.581647e+38
Double
7.29121238410841e-304
Half Float
-55776
String
ÏúíÞ
DOSDATE
06/15/2105
DOSTIME
FILETIME
05/06/1829 01:23:05
Inspector
Variables
Selected: 4 bytes (Range: 0 to 3)
Start: 0 [0h] Sel: 4 [4h] Size: 33,510
Hex ANSI LIT
OVR

Signatures

			tsp fon efi	
`5A 4D`	`ZM`	0	exe	DOS ZM executable and its descendants (rare)

`50 4B 03 04`
`50 4B 05 06` (empty archive)
`50 4B 07 08` (spanned archive)
	`PK␃␄`
	`PK␅␆`
	`PK␇␈`
		0	zip
			aar
			apk
			docx
			epub
			ipa
			jar
			kmz
			maff
			msix
			odp
			ods
			odt
			pk3
			pk4
			pptx
			usdz
			vsdx
			xlsx
			xpi
				zip file format and formats based on it, such as EPUB, JAR, ODF, OOXML

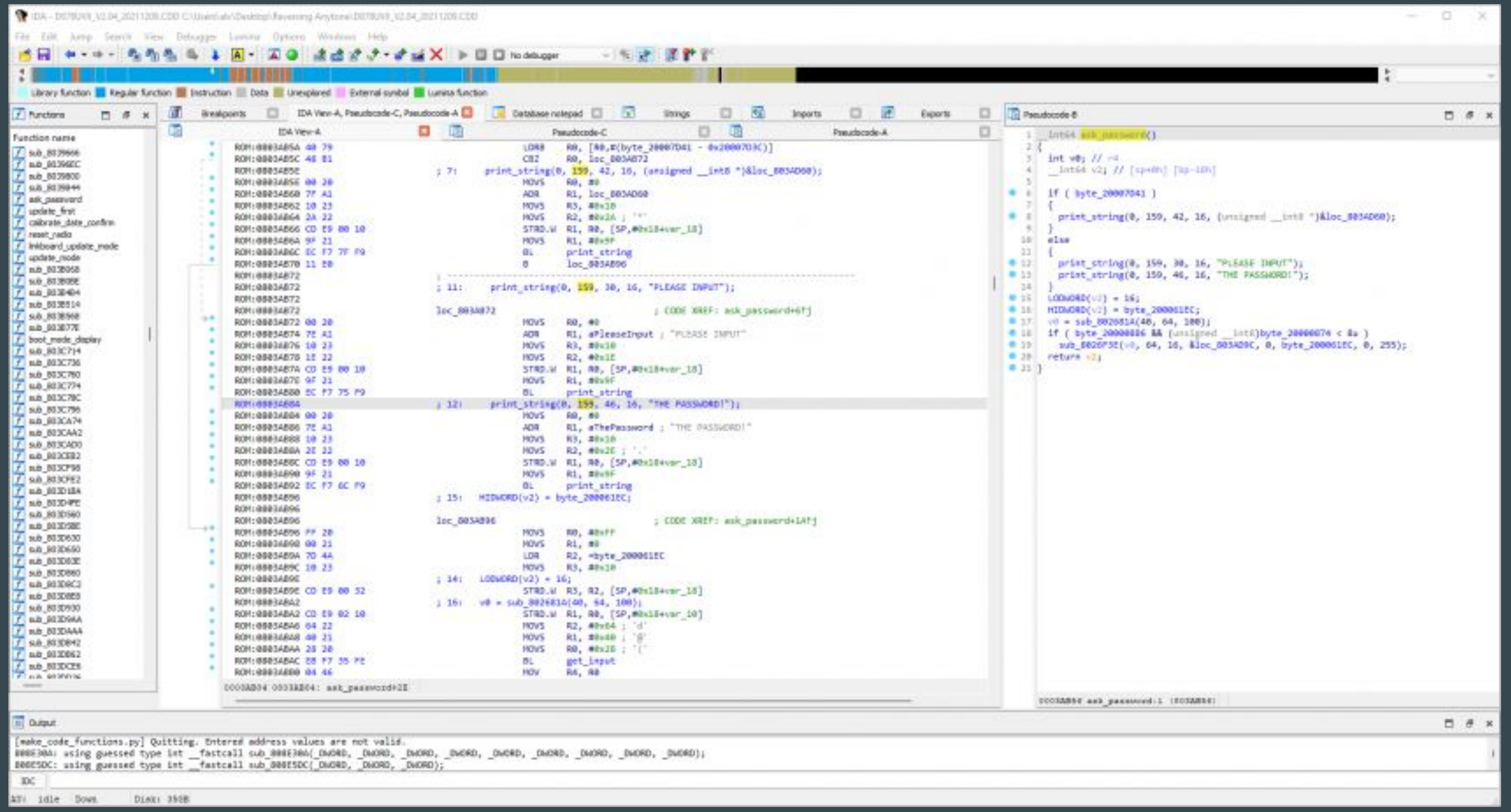
`52 61 72 21 1A 07 00`	`Rar!␚␇␀`	0	rar	Roshal ARchive compressed archive v1.50 onwards[19]
`52 61 72 21 1A 07 01 00`	`Rar!␚␇␁␀`	0	rar	Roshal ARchive compressed archive v5.00 onwards[20]
`7F 45 4C 46`	`␡ELF`	0		Executable and Linkable Format

https://en.wikipedia.org/wiki/List_of_file_signatures

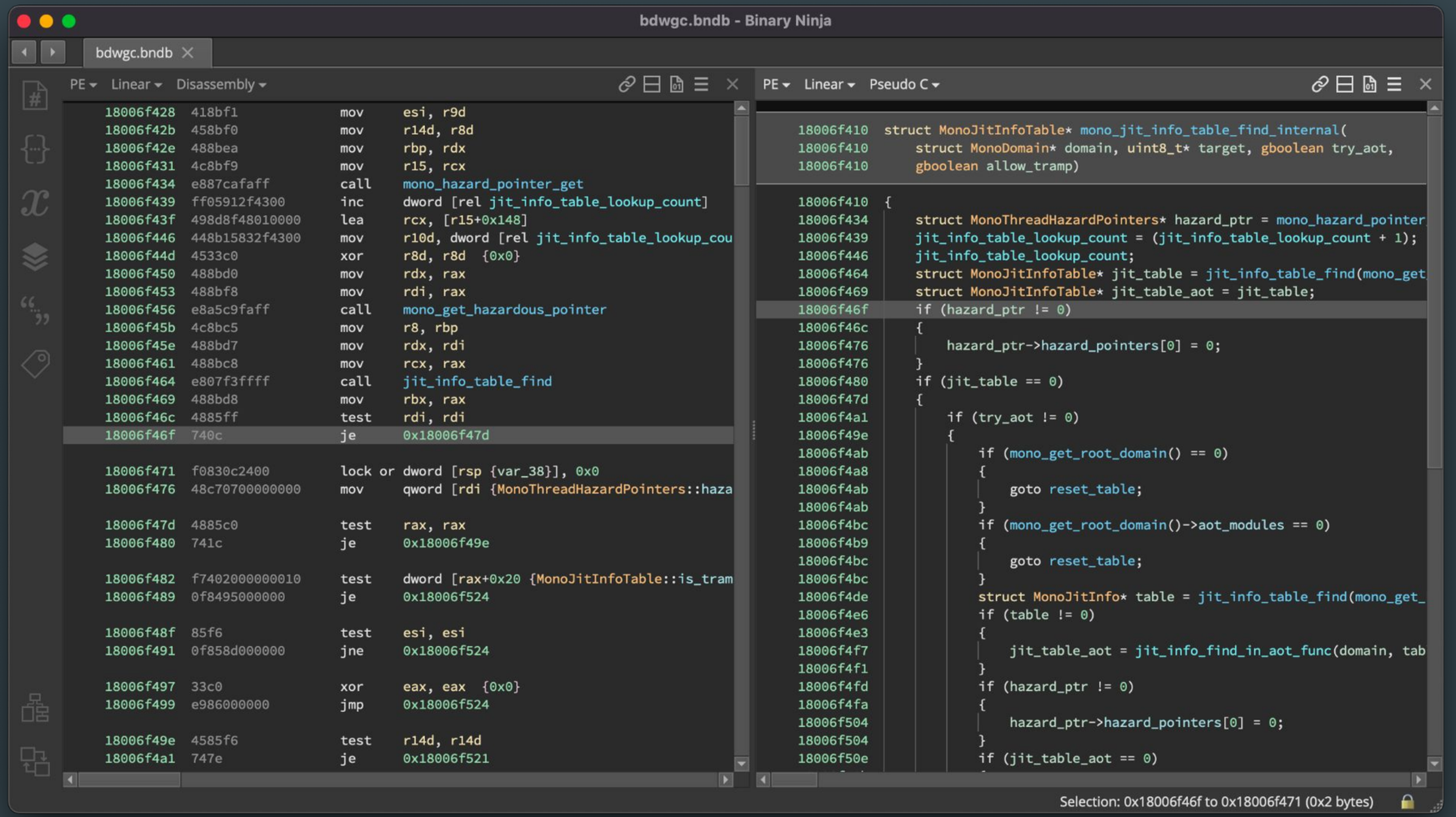
Change hex in fuzzing process

https://en.wikipedia.org/wiki/American_fuzzy_lop_(fuzzer)#/media/File:AFL_Fuzz_Logo.gif

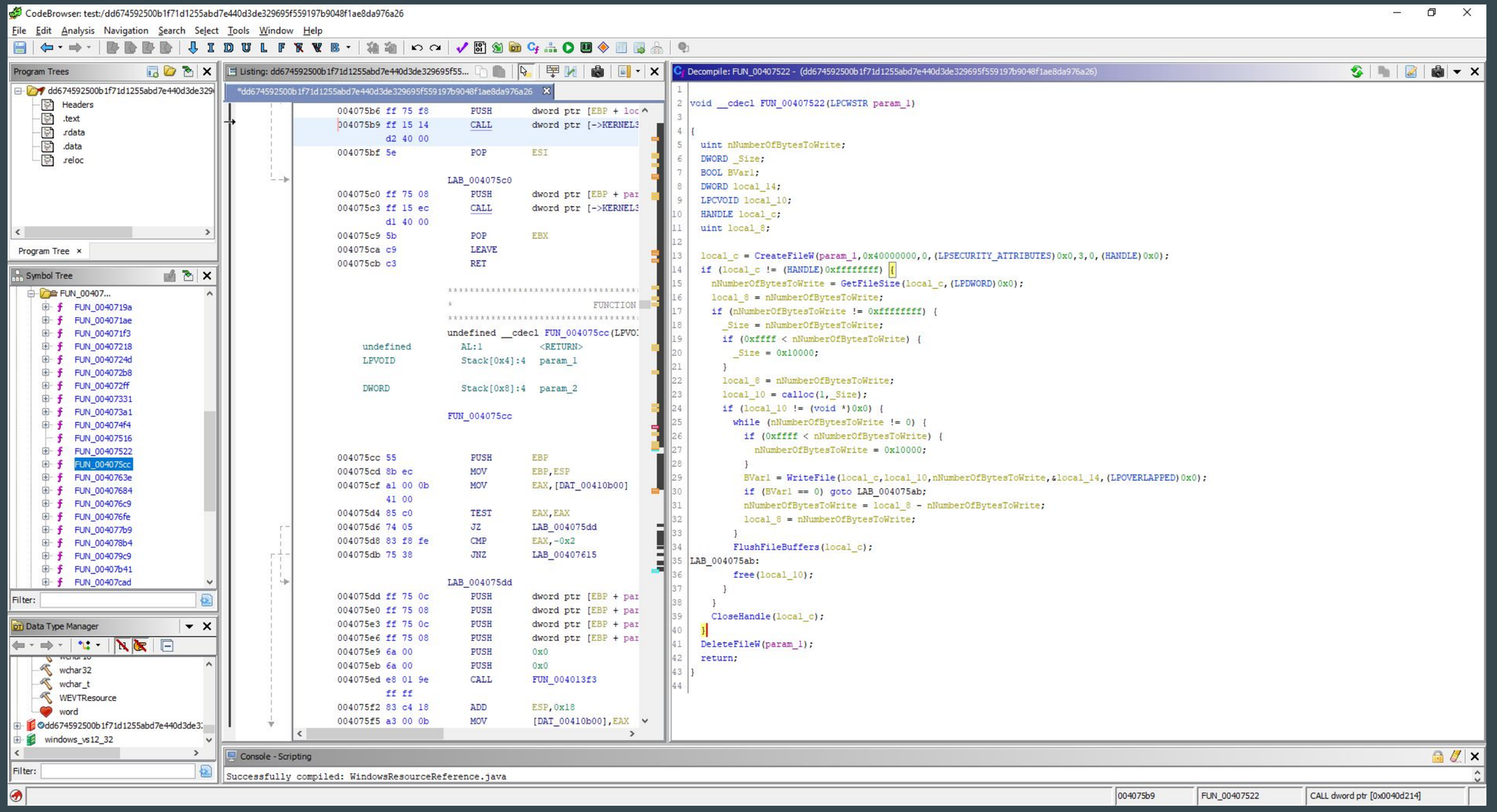
IDA


Functions
Function name
sub_8039666
sub_80396EC
sub_8039800
sub_8039844
ask_password
update_first
calibrate_date_confirm
reset_radio
keyboard_update_mode
update_mode
sub_803B068
sub_803B0BE
sub_803B404
sub_803B514
sub_803B568
sub_803B77E
boot_mode_display
sub_803C714
sub_803C736
sub_803C760
sub_803C774
sub_803C78C
sub_803C796
sub_803CA74
sub_803CAA2
sub_803CAD0
sub_803CEB2
sub_803CF98
sub_803CFE2
sub_803D18A
sub_803D4FE
sub_803D560
sub_803D5BE
sub_803D630
sub_803D650
sub_803D63E
sub_803D680
sub_803D6C2
sub_803D8E8
sub_803D930
sub_803D94A
sub_803DAAA
sub_803DB42
sub_803DB62
sub_803DCE8
Breakpoints
IDA View-A, Pseudocode-C, Pseudocode-A
Database notepad
Strings
Imports
Exports
IDA View-A
Pseudocode-C
Pseudocode-A
ROM:0803A85A 40 79 LDRB R0, [R0,#(byte_20007D41 - 0x20007D3C)]
ROM:0803A85C 48 B1 CBZ R0, loc_803A872
ROM:0803A85E ; 7: print_string(0, 159, 42, 10, (unsigned __int8 *)&loc_803AD60);
ROM:0803A85E 00 20 MOVS R0, #0
ROM:0803A860 7F A1 ADR R1, loc_803AD60
ROM:0803A862 10 23 MOVS R3, #0x10
ROM:0803A864 2A 22 MOVS R2, #0x2A ; '*'
ROM:0803A866 CD E9 00 10 STRD.W R1, R0, [SP,#0x184+var_18]
ROM:0803A86A 9F 21 MOVS R1, #0x9F
ROM:0803A86C DC F7 7F F9 BL print_string
ROM:0803A870 11 E0 B loc_803A896
ROM:0803A872 ; 11: print_string(0, 159, 30, 16, "PLEASE INPUT");
ROM:0803A872 loc_803A872 ; CODE XREF: ask_password+6Fj
ROM:0803A872 00 20 MOVS R0, #0
ROM:0803A874 7E A1 ADR R1, aPleaseInput ; "PLEASE INPUT"
ROM:0803A876 10 23 MOVS R3, #0x10
ROM:0803A878 1E 22 MOVS R2, #0x1E
ROM:0803A87A CD E9 00 10 STRD.W R1, R0, [SP,#0x184+var_18]
ROM:0803A87E 9F 21 MOVS R1, #0x9F
ROM:0803A880 DC F7 75 F9 BL print_string
ROM:0803A884 ; 12: print_string(0, 159, 46, 16, "THE PASSWORD!");
ROM:0803A884 00 20 MOVS R0, #0
ROM:0803A886 7E A1 ADR R1, aThePassword ; "THE PASSWORD!"
ROM:0803A888 10 23 MOVS R3, #0x10
ROM:0803A88A 2E 22 MOVS R2, #0x2E ; '.'
ROM:0803A88C CD E9 00 10 STRD.W R1, R0, [SP,#0x184+var_18]
ROM:0803A890 9F 21 MOVS R1, #0x9F
ROM:0803A892 DC F7 6C F9 BL print_string
ROM:0803A896 ; 15: HIDWORD(v2) = byte_200061EC;
ROM:0803A896 loc_803A896 ; CODE XREF: ask_password+1Af
ROM:0803A896 FF 20 MOVS R0, #0xFF
ROM:0803A898 00 21 MOVS R1, #0
ROM:0803A89A 7D 4A LDR R2, =byte_200061EC
ROM:0803A89C 10 23 MOVS R3, #0x10
ROM:0803A89E ; 14: LODWORD(v2) = 16;
ROM:0803A89E CD E9 00 32 STRD.W R3, R2, [SP,#0x184+var_18]
ROM:0803ABA2 ; 16: v0 = sub_8026814(40, 64, 100);
ROM:0803ABA2 CD E9 02 10 STRD.W R1, R0, [SP,#0x184+var_10]
ROM:0803ABA6 64 22 MOVS R2, #0x64 ; 'd'
ROM:0803ABA8 40 21 MOVS R1, #0x40 ; '@'
ROM:0803ABAA 28 20 MOVS R0, #0x28 ; '('
ROM:0803ABAC 88 F7 35 FE BL get_input
ROM:0803ABB0 04 46 MOV R4, R0
0003AB04 0803AB04: ask_password+2E
Pseudocode-B
1 __int64 ask_password()
2 {
3 int v0; // r4
4 __int64 v2; // [sp+0h] [bp-10h]
5
6 if (byte_20007D41)
7 {
8 print_string(0, 159, 42, 16, (unsigned __int8 *)&loc_803AD60);
9 }
10 else
11 {
12 print_string(0, 159, 30, 16, "PLEASE INPUT");
13 print_string(0, 159, 46, 16, "THE PASSWORD!");
14 }
15 LODWORD(v2) = 16;
16 HIDWORD(v2) = byte_200061EC;
17 v0 = sub_8026814(40, 64, 100);
18 if (byte_20000886 && (unsigned __int8)byte_20000874 < 8u)
19 sub_8026F3E(v0, 64, 16, &loc_803AD0C, 0, byte_200061EC, 0, 255);
20 return v2;
21 }
Output
[make_code_functions.py] Quitting. Entered address values are not valid.
8085E30A: using guessed type int __fastcall sub_8085E30A(_DWORD, _DWORD, _DWORD, _DWORD, _DWORD, _DWORD, _DWORD, _DWORD, _DWORD);
8085E5DC: using guessed type int __fastcall sub_8085E5DC(_DWORD, _DWORD, _DWORD);
IDC
AT: idle Down Disk: 39GB

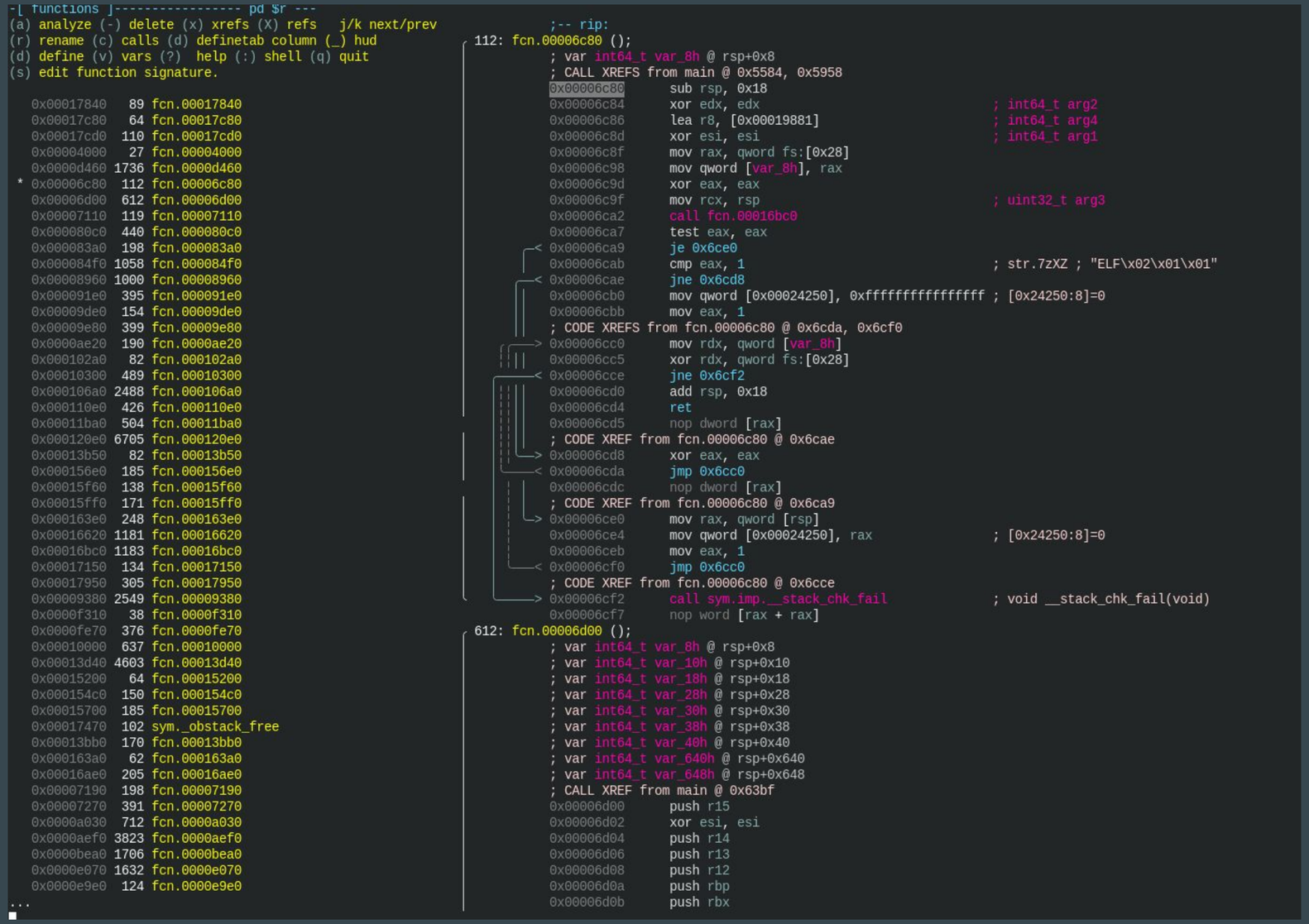
Binary Ninja



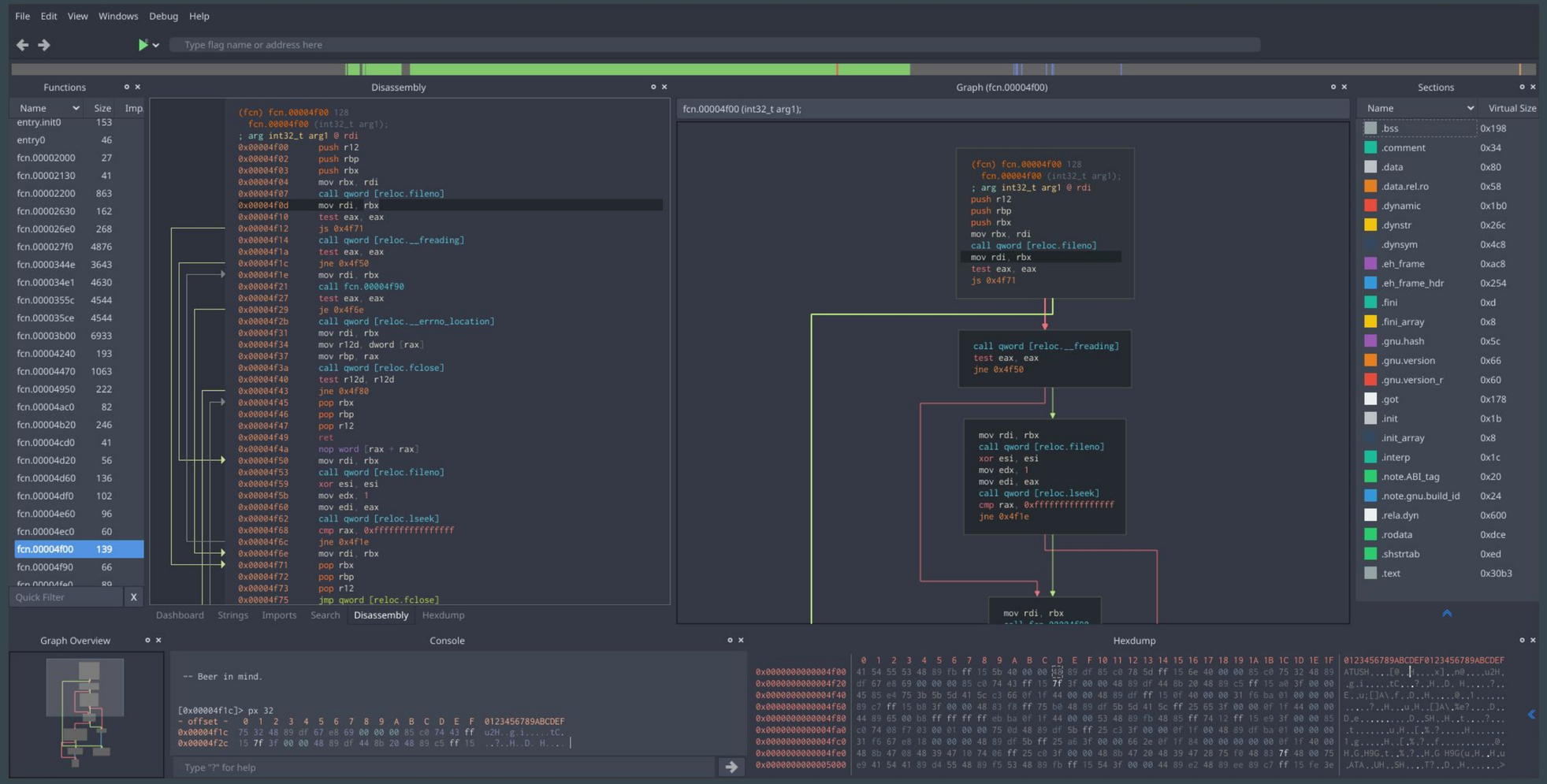
Ghidra


CodeBrowser: test:/dd674592500b1f71d1255abd7e440d3de329695f559197b9048f1ae8da976a26
File Edit Analysis Navigation Search Select Tools Window Help
Program Trees
Symbol Tree
Data Type Manager
Listing: dd674592500b1f71d1255abd7e440d3de329695f55...
Decompile: FUN_00407522 - (dd674592500b1f71d1255abd7e440d3de329695f559197b9048f1ae8da976a26)
Console - Scripting
Successfully compiled: WindowsResourceReference.java

radare2 -> rizin



Cutter


File Edit View Windows Debug Help
Type flag name or address here
Functions
Name Size Imp
entry.init0 153
entry0 46
fcn.00002000 27
fcn.00002130 41
fcn.00002200 863
fcn.00002630 162
fcn.000026e0 268
fcn.000027f0 4876
fcn.0000344e 3643
fcn.000034e1 4630
fcn.0000355c 4544
fcn.000035ce 4544
fcn.00003b00 6933
fcn.00004240 193
fcn.00004470 1063
fcn.00004950 222
fcn.00004ac0 82
fcn.00004b20 246
fcn.00004cd0 41
fcn.00004d20 56
fcn.00004d60 136
fcn.00004df0 102
fcn.00004e60 96
fcn.00004ec0 60
fcn.00004f00 139
fcn.00004f90 66
Quick Filter
Disassembly
(fcn) fcn.00004f00 128
fcn.00004f00 (int32_t arg1);
; arg int32_t arg1 @ rdi
0x00004f00 push r12
0x00004f02 push rbp
0x00004f03 push rbx
0x00004f04 mov rbx, rdi
0x00004f07 call qword [reloc.fileno]
0x00004f0d mov rdi, rbx
0x00004f10 test eax, eax
0x00004f12 js 0x4f71
0x00004f14 call qword [reloc.__freading]
0x00004f1a test eax, eax
0x00004f1c jne 0x4f50
0x00004f1e mov rdi, rbx
0x00004f21 call fcn.00004f90
0x00004f27 test eax, eax
0x00004f29 je 0x4f6e
0x00004f2b call qword [reloc.__errno_location]
0x00004f31 mov rdi, rbx
0x00004f34 mov r12d, dword [rax]
0x00004f37 mov rbp, rax
0x00004f3a call qword [reloc.fclose]
0x00004f40 test r12d, r12d
0x00004f43 jne 0x4f80
0x00004f45 pop rbx
0x00004f46 pop rbp
0x00004f47 pop r12
0x00004f49 ret
0x00004f4a nop word [rax + rax]
0x00004f50 mov rdi, rbx
0x00004f53 call qword [reloc.fileno]
0x00004f59 xor esi, esi
0x00004f5b mov edx, 1
0x00004f60 mov edi, eax
0x00004f62 call qword [reloc.lseek]
0x00004f68 cmp rax, 0xffffffffffffffff
0x00004f6c jne 0x4f1e
0x00004f6e mov rdi, rbx
0x00004f71 pop rbx
0x00004f72 pop rbp
0x00004f73 pop r12
0x00004f75 jmp qword [reloc.fclose]
Dashboard Strings Imports Search Disassembly Hexdump
Graph (fcn.00004f00)
fcn.00004f00 (int32_t arg1);
(fcn) fcn.00004f00 128
fcn.00004f00 (int32_t arg1);
; arg int32_t arg1 @ rdi
push r12
push rbp
push rbx
mov rbx, rdi
call qword [reloc.fileno]
mov rdi, rbx
test eax, eax
js 0x4f71
call qword [reloc.__freading]
test eax, eax
jne 0x4f50
mov rdi, rbx
call qword [reloc.fileno]
xor esi, esi
mov edx, 1
mov edi, eax
call qword [reloc.lseek]
cmp rax, 0xffffffffffffffff
jne 0x4f1e
mov rdi, rbx
Sections
Name Virtual Size
.bss 0x198
.comment 0x34
.data 0x80
.data.rel.ro 0x58
.dynamic 0x1b0
.dynstr 0x26c
.dynsym 0x4c8
.eh_frame 0xac8
.eh_frame_hdr 0x254
.fini 0xd
.fini_array 0x8
.gnu.hash 0x5c
.gnu.version 0x66
.gnu.version_r 0x60
.got 0x178
.init 0x1b
.init_array 0x8
.interp 0x1c
.note.ABI_tag 0x20
.note.gnu.build_id 0x24
.rela.dyn 0x600
.rodata 0xdce
.shstrtab 0xed
.text 0x30b3
Graph Overview
Console
-- Beer in mind.
[0x00004f1c]> px 32
- offset - 0 1 2 3 4 5 6 7 8 9 A B C D E F 0123456789ABCDEF
0x00004f1c 75 32 48 89 df 67 e8 69 00 00 00 85 c0 74 43 ff u2H..g.i.....tC.
0x00004f2c 15 7f 3f 00 00 48 89 df 44 8b 20 48 89 c5 ff 15 ..?..H..D. H....
Type "?" for help
Hexdump
0 1 2 3 4 5 6 7 8 9 A B C D E F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F 0123456789ABCDEF0123456789ABCDEF
0x0000000000004f00 41 54 55 53 48 89 fb ff 15 5b 40 00 00 89 df 85 c0 78 5d ff 15 6e 40 00 00 85 c0 75 32 48 89 ATUSH....[@.....x]..n@....u2H.
0x0000000000004f20 df 67 e8 69 00 00 00 85 c0 74 43 ff 15 7f 3f 00 00 48 89 df 44 8b 20 48 89 c5 ff 15 a0 3f 00 00 .g.i.....tC...?..H..D. H.....?..
0x0000000000004f40 45 85 e4 75 3b 5b 5d 41 5c c3 66 0f 1f 44 00 00 48 89 df ff 15 0f 40 00 00 31 f6 ba 01 00 00 00 E..u;[]A\.f..D..H.....@..1.....
0x0000000000004f60 89 c7 ff 15 b8 3f 00 00 48 83 f8 ff 75 b0 48 89 df 5b 5d 41 5c ff 25 65 3f 00 00 0f 1f 44 00 00 .....?..H...u.H..[]A\.%e?....D..
0x0000000000004f80 44 89 65 00 b8 ff ff ff ff eb ba 0f 1f 44 00 00 53 48 89 fb 48 85 ff 74 12 ff 15 e9 3f 00 00 85 D.e.........D..SH..H..t....?...
0x0000000000004fa0 c0 74 08 f7 03 00 01 00 00 75 0d 48 89 df 5b ff 25 c3 3f 00 00 0f 1f 00 48 89 df ba 01 00 00 00 .t.......u.H..[.%.?.....H.......
0x0000000000004fc0 31 f6 67 e8 18 00 00 00 48 89 df 5b ff 25 a6 3f 00 00 66 2e 0f 1f 84 00 00 00 00 00 0f 1f 40 00 1.g.....H..[.%.?..f..........@.
0x0000000000004fe0 48 8b 47 08 48 39 47 10 74 06 ff 25 c0 3f 00 00 48 8b 47 20 48 39 47 28 75 f0 48 83 7f 48 00 75 H.G.H9G.t..%.?..H.G H9G(u.H..H.u
0x0000000000005000 e9 41 54 41 89 d4 55 48 89 f5 53 48 89 fb ff 15 54 3f 00 00 44 89 e2 48 89 ee 89 c7 ff 15 fe 3e .ATA..UH..SH....T?..D..H.......>
27

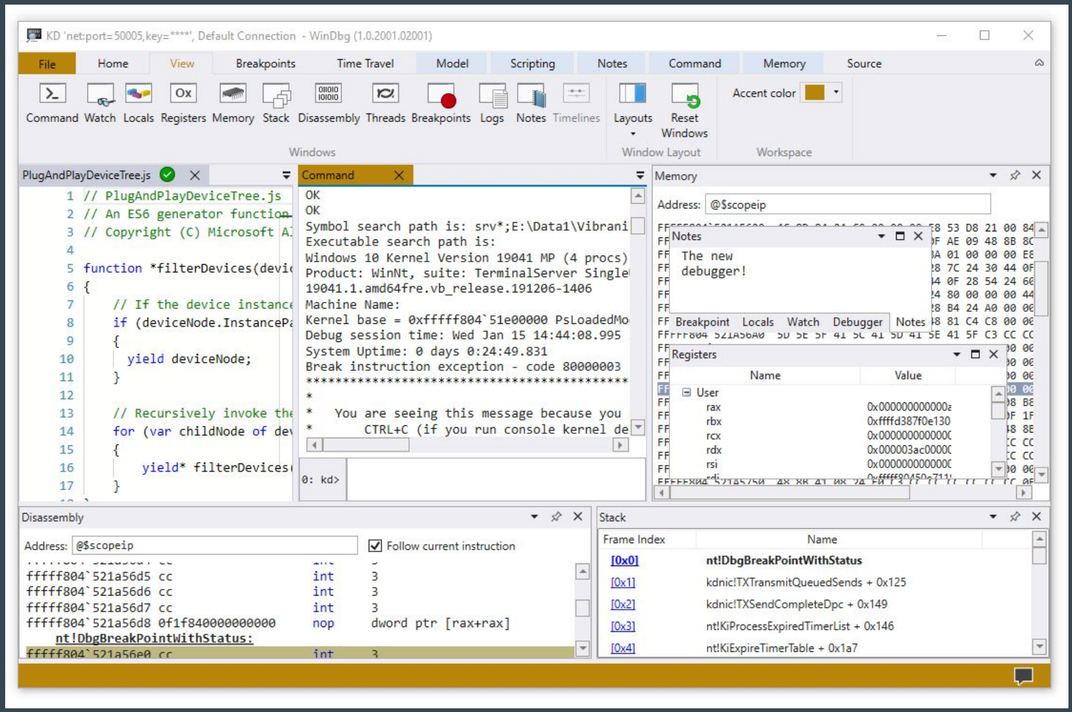
Cutter

```
void sym.Aeropause(Bright *bright, int32_t argc, char **argv)
{
    Morning *pMVar1;
    int32_t iVar2;

    pMVar1 = (Morning *)sym.imp.malloc(8);
    bright->morning = pMVar1;
    bright->morning->saved_argc = argc;
    bright->morning->saved_argv = argv;
    if (bright->morning->saved_argc < 2) {
        bright->ambassador = AMBASSADOR_PURE;
    } else {
        (bright->window).sunlight = bright->morning->saved_argv[1];
        iVar2 = strcmp((bright->window).sunlight, "the  ");
        if (iVar2 == 0) {
            bright->ambassador = AMBASSADOR_REASON;
        } else {
            iVar2 = strcmp((bright->window).sunlight, "dark");
            if (iVar2 == 0) {
                bright->ambassador = AMBASSADOR_REVOLUTION;
            } else {
                iVar2 = strcmp((bright->window).sunlight, "third");
```

Debuggers

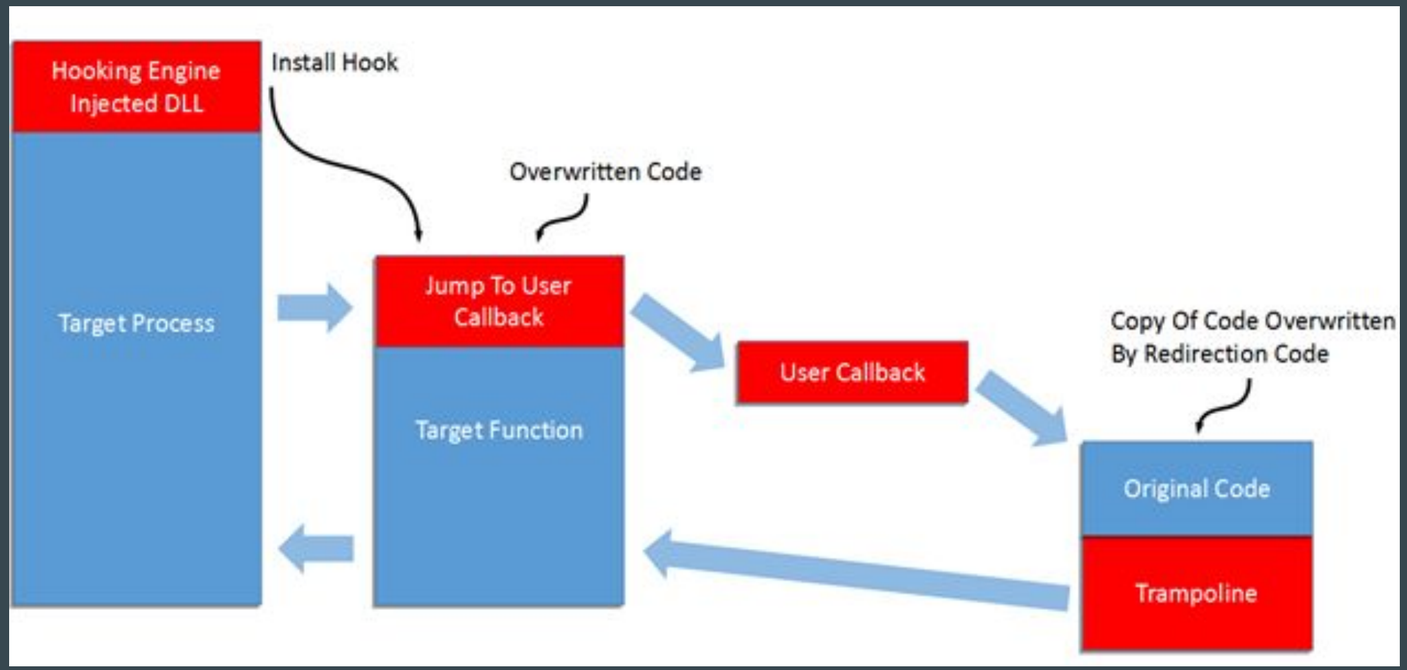
- windbg
- gdb / lldb
- embedded




```
[ Legend: Modified register | Code | Heap | Stack | String ]
registers
$rax   : 0x0
$rbx   : 0x0
$rcx   : 0x00007ffff7ffcca0  →  0x0004095d00000000
$rdx   : 0x0
$rsp   : 0x00007fffffffe530  →  0x0000000000000000
$rbp   : 0x00007fffffffe560  →  0x00000000004007f0  →  <__libc_csu_init+0> push r15
$rsi   : 0x00007ffff7dd1b78  →  0x0000000000602000  →  0x0000000000000000
$rdi   : 0x20000
$rip   : 0x0000000000400799  →  <main+64> mov QWORD PTR [rbp-0x28], rax
$r8    : 0x00007ffff7fec700  →  0x00007ffff7fec700  →  [loop detected]
$r9    : 0x1
$r10   : 0x0
$r11   : 0x246
$r12   : 0x0000000000400580  →  <_start+0> xor ebp, ebp
$r13   : 0x00007fffffffe640  →  0x0000000000000001
$r14   : 0x0
$r15   : 0x0
$eflags: [carry PARITY adjust ZERO sign trap INTERRUPT direction overflow resume virtualx86 identification]
$ss: 0x002b  $cs: 0x0033  $ds: 0x0000  $gs: 0x0000  $es: 0x0000  $fs: 0x0000
stack
0x00007fffffffe530|+0x0000: 0x0000000000000000   ← $rsp
0x00007fffffffe538|+0x0008: 0x0000000000000000
0x00007fffffffe540|+0x0010: "myfile.txt"
0x00007fffffffe548|+0x0018: 0x0000000000007478 ("xt"?)
0x00007fffffffe550|+0x0020: 0x00007fffffffe640  →  0x0000000000000001
0x00007fffffffe558|+0x0028: 0xd7c3f14d3cddb000
0x00007fffffffe560|+0x0030: 0x00000000004007f0  →  <__libc_csu_init+0> push r15  ← $rbp
0x00007fffffffe568|+0x0038: 0x00007ffff7a2d830  →  <__libc_start_main+240> mov edi, eax
code:i386:x86-64
    0x40078c <main+51>        mov    esi, 0x400874
    0x400791 <main+56>        mov    rdi, rax
    0x400794 <main+59>        call   0x400550 <fopen@plt>
→   0x400799 <main+64>        mov    QWORD PTR [rbp-0x28], rax
    0x40079d <main+68>        cmp    QWORD PTR [rbp-0x28], 0x0
    0x4007a2 <main+73>        jne    0x4007bc <main+99>
    0x4007a4 <main+75>        lea    rax, [rbp-0x20]
    0x4007a8 <main+79>        mov    rsi, rax
    0x4007ab <main+82>        mov    edi, 0x400876
source:vsnprintf.c+20
    15  int main ()
    16  {
    17     FILE * pFile;
    18     char szFileName[]="myfile.txt";
    19
                // pFile=0x00007fffffffe538  →  0x0000000000000000, szFileName=0x00007fffffffe540  →  "myfile.txt"
→   20     pFile = fopen (szFileName,"r");
    21     if (pFile == NULL)
    22       PrintFError ("Error opening '%s'",szFileName);
    23     else
    24     {
    25       // file successfully open
threads
[#0] Id 1, Name: "vsnprintf", stopped, reason: SINGLE STEP
trace
[#0] 0x400799 → Name: main()
gef➤
```

DBI

- PIN
- dynamorio
- frida

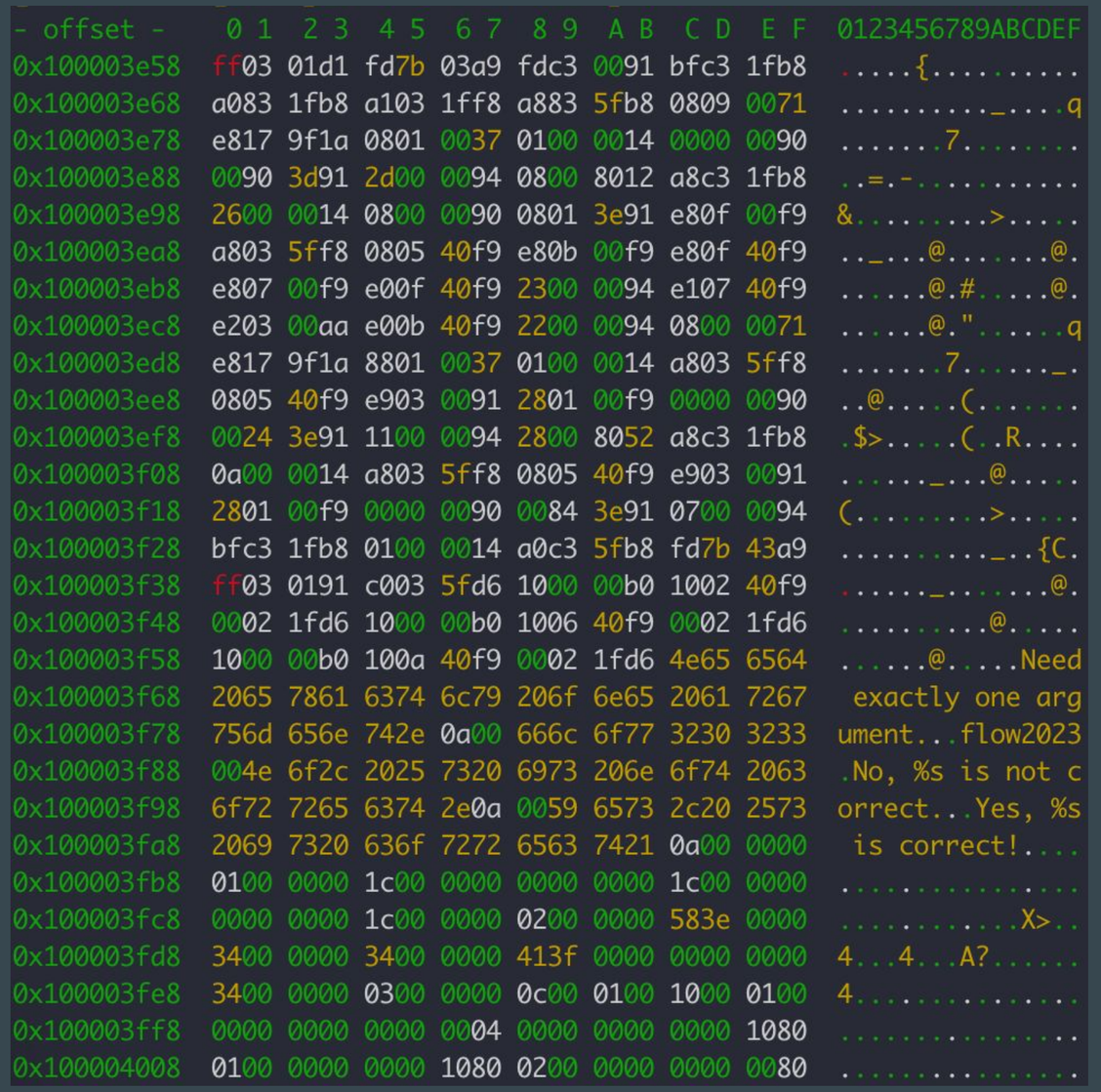


Demo

Demo

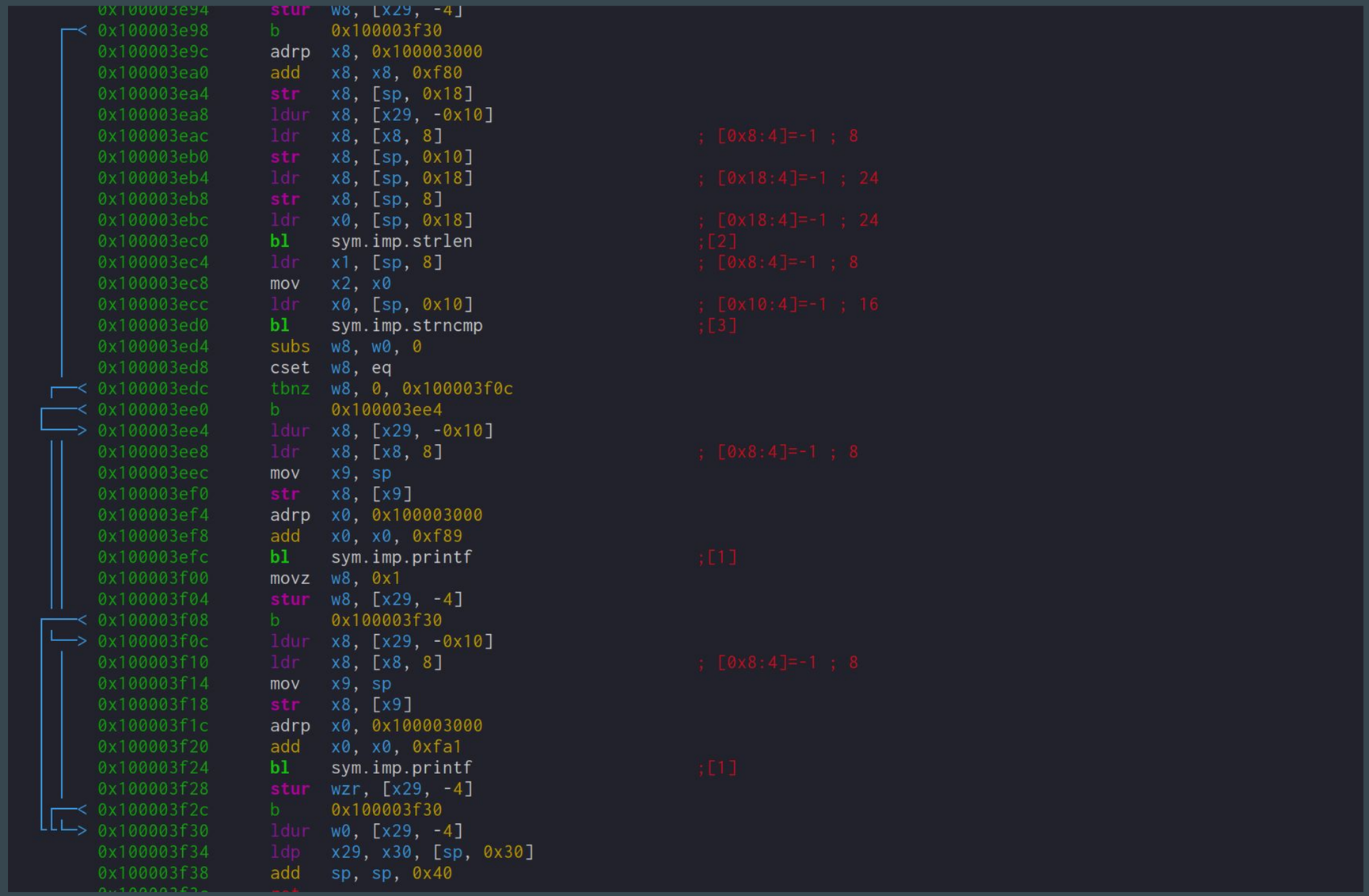
```
 ~/Projects/light_reverse/ ./crackme01
Need exactly one argument.
 ~/Projects/light_reverse/ ./crackme01 qwerty
No, qwerty is not correct.
 ~/Projects/light_reverse/
```

Demo

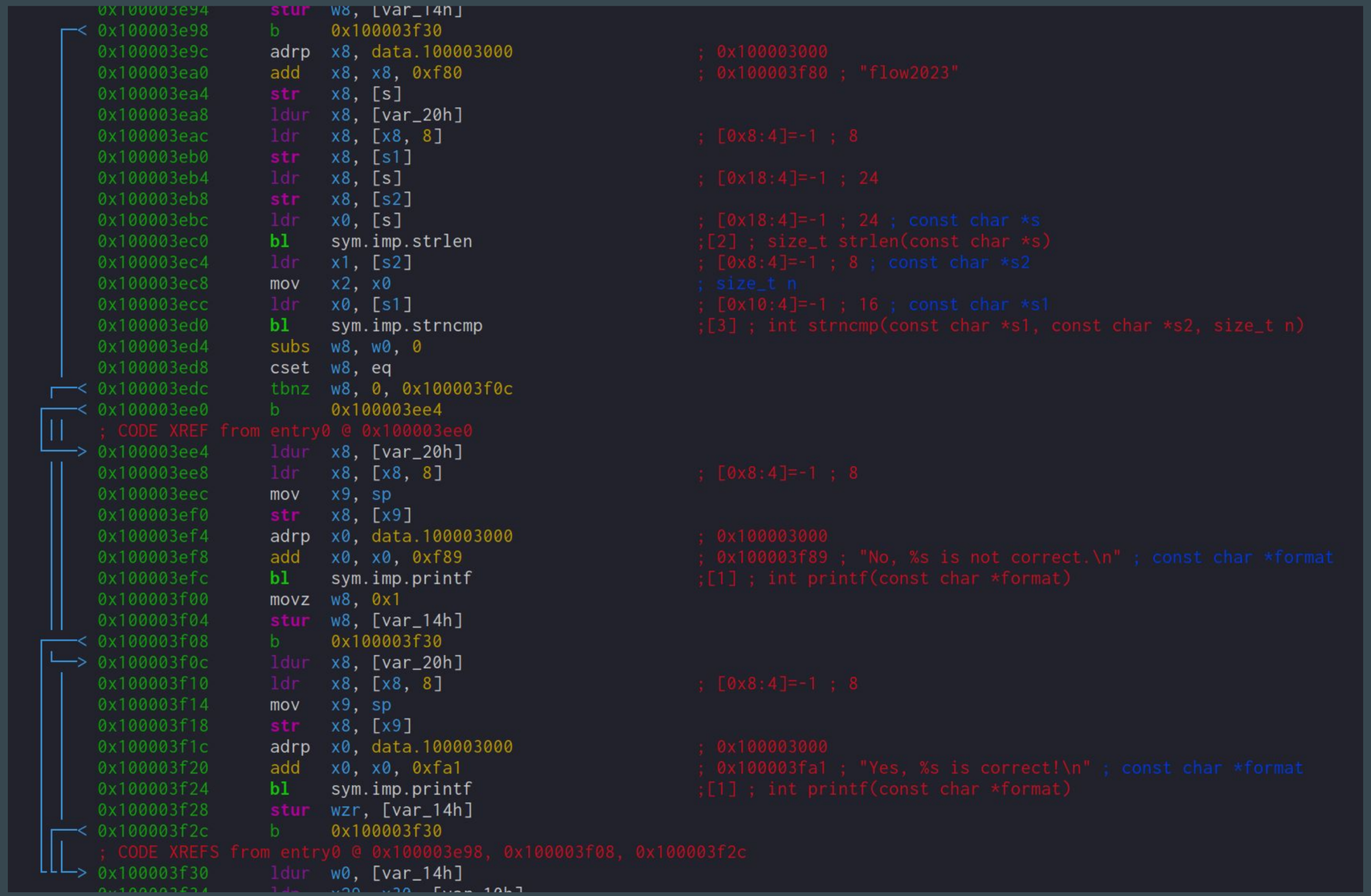


```
- offset -    0 1  2 3  4 5  6 7  8 9  A B  C D  E F  0123456789ABCDEF
0x100003e58  ff03 01d1 fd7b 03a9 fdc3 0091 bfc3 1fb8  .....{..........
0x100003e68  a083 1fb8 a103 1ff8 a883 5fb8 0809 0071  .........._....q
0x100003e78  e817 9f1a 0801 0037 0100 0014 0000 0090  .......7........
0x100003e88  0090 3d91 2d00 0094 0800 8012 a8c3 1fb8  ..=.-...........
0x100003e98  2600 0014 0800 0090 0801 3e91 e80f 00f9  &.........>.....
0x100003ea8  a803 5ff8 0805 40f9 e80b 00f9 e80f 40f9  .._...@.......@.
0x100003eb8  e807 00f9 e00f 40f9 2300 0094 e107 40f9  ......@.#.....@.
0x100003ec8  e203 00aa e00b 40f9 2200 0094 0800 0071  ......@."......q
0x100003ed8  e817 9f1a 8801 0037 0100 0014 a803 5ff8  .......7......_.
0x100003ee8  0805 40f9 e903 0091 2801 00f9 0000 0090  ..@.....(.......
0x100003ef8  0024 3e91 1100 0094 2800 8052 a8c3 1fb8  .$>.....(..R....
0x100003f08  0a00 0014 a803 5ff8 0805 40f9 e903 0091  ......_...@.....
0x100003f18  2801 00f9 0000 0090 0084 3e91 0700 0094  (.........>.....
0x100003f28  bfc3 1fb8 0100 0014 a0c3 5fb8 fd7b 43a9  ..........._..{C.
0x100003f38  ff03 0191 c003 5fd6 1000 00b0 1002 40f9  ......_.......@.
0x100003f48  0002 1fd6 1000 00b0 1006 40f9 0002 1fd6  ..........@.....
0x100003f58  1000 00b0 100a 40f9 0002 1fd6 4e65 6564  ......@.....Need
0x100003f68  2065 7861 6374 6c79 206f 6e65 2061 7267   exactly one arg
0x100003f78  756d 656e 742e 0a00 666c 6f77 3230 3233  ument...flow2023
0x100003f88  004e 6f2c 2025 7320 6973 206e 6f74 2063  .No, %s is not c
0x100003f98  6f72 7265 6374 2e0a 0059 6573 2c20 2573  orrect...Yes, %s
0x100003fa8  2069 7320 636f 7272 6563 7421 0a00 0000   is correct!....
0x100003fb8  0100 0000 1c00 0000 0000 0000 1c00 0000  ................
0x100003fc8  0000 0000 1c00 0000 0200 0000 583e 0000  ............X>..
0x100003fd8  3400 0000 3400 0000 413f 0000 0000 0000  4...4...A?......
0x100003fe8  3400 0000 0300 0000 0c00 0100 1000 0100  4...............
0x100003ff8  0000 0000 0000 0004 0000 0000 0000 1080  ................
0x100004008  0100 0000 0000 1080 0200 0000 0000 0080  ................
```

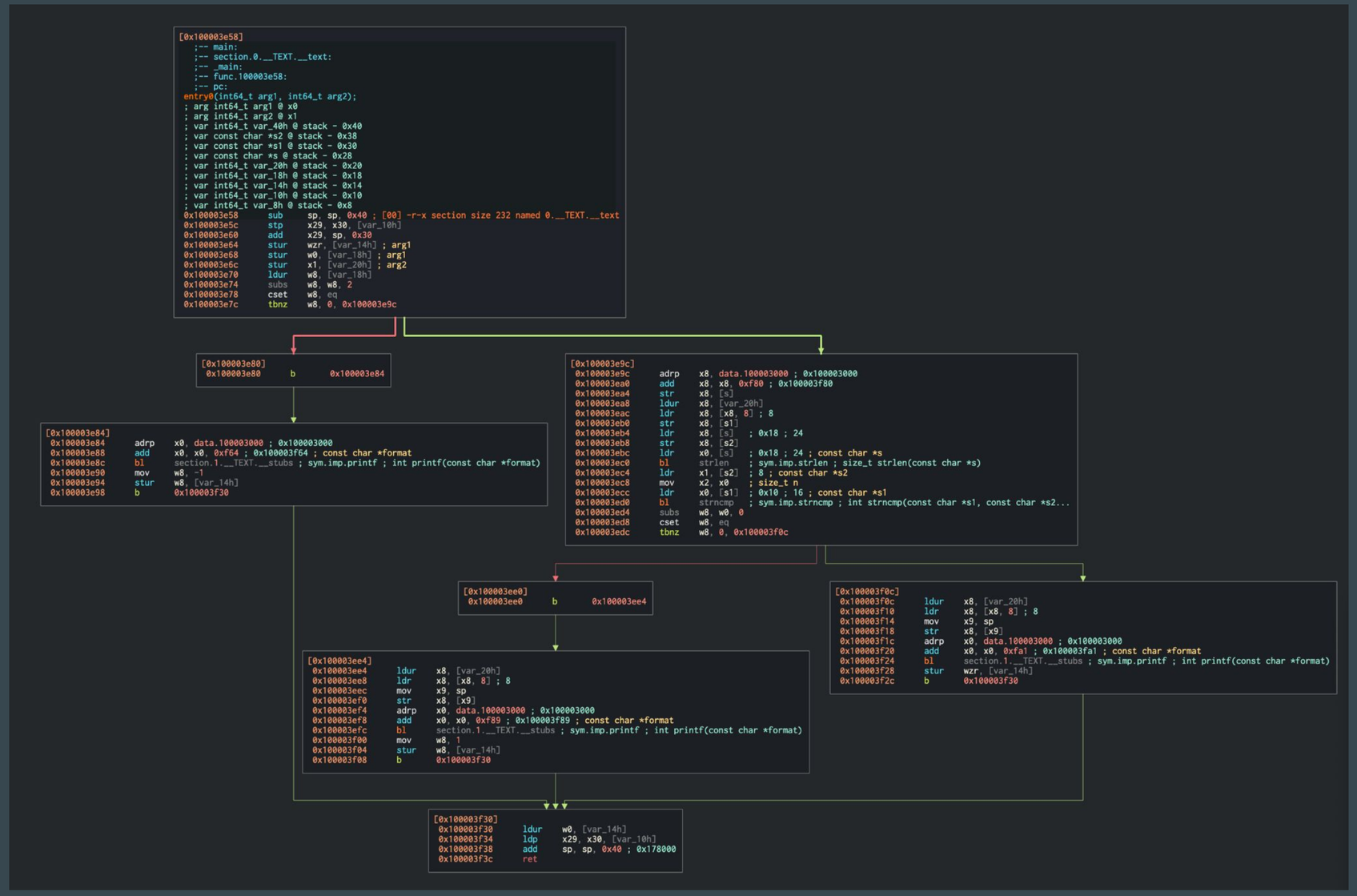
Demo


0x100003e94 stur w8, [x29, -4]
0x100003e98 b 0x100003f30
0x100003e9c adrp x8, 0x100003000
0x100003ea0 add x8, x8, 0xf80
0x100003ea4 str x8, [sp, 0x18]
0x100003ea8 ldur x8, [x29, -0x10]
0x100003eac ldr x8, [x8, 8] ; [0x8:4]=-1 ; 8
0x100003eb0 str x8, [sp, 0x10]
0x100003eb4 ldr x8, [sp, 0x18] ; [0x18:4]=-1 ; 24
0x100003eb8 str x8, [sp, 8]
0x100003ebc ldr x0, [sp, 0x18] ; [0x18:4]=-1 ; 24
0x100003ec0 bl sym.imp.strlen ;[2]
0x100003ec4 ldr x1, [sp, 8] ; [0x8:4]=-1 ; 8
0x100003ec8 mov x2, x0
0x100003ecc ldr x0, [sp, 0x10] ; [0x10:4]=-1 ; 16
0x100003ed0 bl sym.imp.strncmp ;[3]
0x100003ed4 subs w8, w0, 0
0x100003ed8 cset w8, eq
0x100003edc tbnz w8, 0, 0x100003f0c
0x100003ee0 b 0x100003ee4
0x100003ee4 ldur x8, [x29, -0x10]
0x100003ee8 ldr x8, [x8, 8] ; [0x8:4]=-1 ; 8
0x100003eec mov x9, sp
0x100003ef0 str x8, [x9]
0x100003ef4 adrp x0, 0x100003000
0x100003ef8 add x0, x0, 0xf89
0x100003efc bl sym.imp.printf ;[1]
0x100003f00 movz w8, 0x1
0x100003f04 stur w8, [x29, -4]
0x100003f08 b 0x100003f30
0x100003f0c ldur x8, [x29, -0x10]
0x100003f10 ldr x8, [x8, 8] ; [0x8:4]=-1 ; 8
0x100003f14 mov x9, sp
0x100003f18 str x8, [x9]
0x100003f1c adrp x0, 0x100003000
0x100003f20 add x0, x0, 0xfa1
0x100003f24 bl sym.imp.printf ;[1]
0x100003f28 stur wzr, [x29, -4]
0x100003f2c b 0x100003f30
0x100003f30 ldur w0, [x29, -4]
0x100003f34 ldp x29, x30, [sp, 0x30]
0x100003f38 add sp, sp, 0x40

Demo


0x100003e94 stur w8, [var_14h]
0x100003e98 b 0x100003f30
0x100003e9c adrp x8, data.100003000 ; 0x100003000
0x100003ea0 add x8, x8, 0xf80 ; 0x100003f80 ; "flow2023"
0x100003ea4 str x8, [s]
0x100003ea8 ldur x8, [var_20h]
0x100003eac ldr x8, [x8, 8] ; [0x8:4]=-1 ; 8
0x100003eb0 str x8, [s1]
0x100003eb4 ldr x8, [s] ; [0x18:4]=-1 ; 24
0x100003eb8 str x8, [s2]
0x100003ebc ldr x0, [s] ; [0x18:4]=-1 ; 24 ; const char *s
0x100003ec0 bl sym.imp.strlen ;[2] ; size_t strlen(const char *s)
0x100003ec4 ldr x1, [s2] ; [0x8:4]=-1 ; 8 ; const char *s2
0x100003ec8 mov x2, x0 ; size_t n
0x100003ecc ldr x0, [s1] ; [0x10:4]=-1 ; 16 ; const char *s1
0x100003ed0 bl sym.imp.strncmp ;[3] ; int strncmp(const char *s1, const char *s2, size_t n)
0x100003ed4 subs w8, w0, 0
0x100003ed8 cset w8, eq
0x100003edc tbnz w8, 0, 0x100003f0c
0x100003ee0 b 0x100003ee4
; CODE XREF from entry0 @ 0x100003ee0
0x100003ee4 ldur x8, [var_20h]
0x100003ee8 ldr x8, [x8, 8] ; [0x8:4]=-1 ; 8
0x100003eec mov x9, sp
0x100003ef0 str x8, [x9]
0x100003ef4 adrp x0, data.100003000 ; 0x100003000
0x100003ef8 add x0, x0, 0xf89 ; 0x100003f89 ; "No, %s is not correct.\n" ; const char *format
0x100003efc bl sym.imp.printf ;[1] ; int printf(const char *format)
0x100003f00 movz w8, 0x1
0x100003f04 stur w8, [var_14h]
0x100003f08 b 0x100003f30
0x100003f0c ldur x8, [var_20h]
0x100003f10 ldr x8, [x8, 8] ; [0x8:4]=-1 ; 8
0x100003f14 mov x9, sp
0x100003f18 str x8, [x9]
0x100003f1c adrp x0, data.100003000 ; 0x100003000
0x100003f20 add x0, x0, 0xfa1 ; 0x100003fa1 ; "Yes, %s is correct!\n" ; const char *format
0x100003f24 bl sym.imp.printf ;[1] ; int printf(const char *format)
0x100003f28 stur wzr, [var_14h]
0x100003f2c b 0x100003f30
; CODE XREFS from entry0 @ 0x100003e98, 0x100003f08, 0x100003f2c
0x100003f30 ldur w0, [var_14h]

Demo


[0x100003e58]
;-- main:
;-- section.0.__TEXT.__text:
;-- _main:
;-- func.100003e58:
;-- pc:
entry0(int64_t arg1, int64_t arg2);
; arg int64_t arg1 @ x0
; arg int64_t arg2 @ x1
; var int64_t var_40h @ stack - 0x40
; var const char *s2 @ stack - 0x38
; var const char *s1 @ stack - 0x30
; var const char *s @ stack - 0x28
; var int64_t var_20h @ stack - 0x20
; var int64_t var_18h @ stack - 0x18
; var int64_t var_14h @ stack - 0x14
; var int64_t var_10h @ stack - 0x10
; var int64_t var_8h @ stack - 0x8
0x100003e58 sub sp, sp, 0x40 ; [00] -r-x section size 232 named 0.__TEXT.__text
0x100003e5c stp x29, x30, [var_10h]
0x100003e60 add x29, sp, 0x30
0x100003e64 stur wzr, [var_14h] ; arg1
0x100003e68 stur w0, [var_18h] ; arg1
0x100003e6c stur x1, [var_20h] ; arg2
0x100003e70 ldur w8, [var_18h]
0x100003e74 subs w8, w8, 2
0x100003e78 cset w8, eq
0x100003e7c tbnz w8, 0, 0x100003e9c
[0x100003e80]
0x100003e80 b 0x100003e84
[0x100003e84]
0x100003e84 adrp x0, data.100003000 ; 0x100003000
0x100003e88 add x0, x0, 0xf64 ; 0x100003f64 ; const char *format
0x100003e8c bl section.1.__TEXT.__stubs ; sym.imp.printf ; int printf(const char *format)
0x100003e90 mov w8, -1
0x100003e94 stur w8, [var_14h]
0x100003e98 b 0x100003f30
[0x100003e9c]
0x100003e9c adrp x8, data.100003000 ; 0x100003000
0x100003ea0 add x8, x8, 0xf80 ; 0x100003f80
0x100003ea4 str x8, [s]
0x100003ea8 ldur x8, [var_20h]
0x100003eac ldr x8, [x8, 8] ; 8
0x100003eb0 str x8, [s1]
0x100003eb4 ldr x8, [s] ; 0x18 ; 24
0x100003eb8 str x8, [s2]
0x100003ebc ldr x0, [s] ; 0x18 ; 24 ; const char *s
0x100003ec0 bl strlen ; sym.imp.strlen ; size_t strlen(const char *s)
0x100003ec4 ldr x1, [s2] ; 8 ; const char *s2
0x100003ec8 mov x2, x0 ; size_t n
0x100003ecc ldr x0, [s1] ; 0x10 ; 16 ; const char *s1
0x100003ed0 bl strncmp ; sym.imp.strncmp ; int strncmp(const char *s1, const char *s2...
0x100003ed4 subs w8, w0, 0
0x100003ed8 cset w8, eq
0x100003edc tbnz w8, 0, 0x100003f0c
[0x100003ee0]
0x100003ee0 b 0x100003ee4
[0x100003ee4]
0x100003ee4 ldur x8, [var_20h]
0x100003ee8 ldr x8, [x8, 8] ; 8
0x100003eec mov x9, sp
0x100003ef0 str x8, [x9]
0x100003ef4 adrp x0, data.100003000 ; 0x100003000
0x100003ef8 add x0, x0, 0xf89 ; 0x100003f89 ; const char *format
0x100003efc bl section.1.__TEXT.__stubs ; sym.imp.printf ; int printf(const char *format)
0x100003f00 mov w8, 1
0x100003f04 stur w8, [var_14h]
0x100003f08 b 0x100003f30
[0x100003f0c]
0x100003f0c ldur x8, [var_20h]
0x100003f10 ldr x8, [x8, 8] ; 8
0x100003f14 mov x9, sp
0x100003f18 str x8, [x9]
0x100003f1c adrp x0, data.100003000 ; 0x100003000
0x100003f20 add x0, x0, 0xfa1 ; 0x100003fa1 ; const char *format
0x100003f24 bl section.1.__TEXT.__stubs ; sym.imp.printf ; int printf(const char *format)
0x100003f28 stur wzr, [var_14h]
0x100003f2c b 0x100003f30
[0x100003f30]
0x100003f30 ldur w0, [var_14h]
0x100003f34 ldp x29, x30, [var_10h]
0x100003f38 add sp, sp, 0x40 ; 0x178000
0x100003f3c ret

Demo

```
undefined4 entry0(int64_t arg1, int64_t arg2)
{
    int32_t iVar1;
    undefined8 uVar2;
    undefined8 uVar3;
    int64_t var_40h;
    char *s2;
    char *s1;
    char *s;
    int64_t var_20h;
    undefined4 var_14h;
    int64_t var_10h;
    int64_t var_8h;

    // [00] -r-x section size 232 named 0.__TEXT.__text
    if ((int32_t)arg1 == 2) {
        uVar3 = *(undefined8 *)(arg2 + 8);
        uVar2 = strlen("flow2023");
        iVar1 = strncmp(uVar3, "flow2023", uVar2);
        if (iVar1 == 0) {
            printf("Yes, %s is correct!\n");
            var_14h = 0;
        } else {
            printf("No, %s is not correct.\n");
            var_14h = 1;
        }
    } else {
        printf("Need exactly one argument.\n");
        var_14h = 0xffffffff;
    }
    return var_14h;
}
```

```c
#include <stdio.h>
#include <string.h>

// A very simple crackme which stores the correct password in program memory
// and uses the builtin string comparison function to check it.

int main(int argc, char** argv) {

    if (argc != 2) {
        printf("Need exactly one argument.\n");
        return -1;
    }

    char* correct = "flow2023";

    if (strncmp(argv[1], correct, strlen(correct))) {
        printf("No, %s is not correct.\n", argv[1]);
        return 1;
    } else {
        printf("Yes, %s is correct!\n", argv[1]);
        return 0;
    }

}
```

Demo

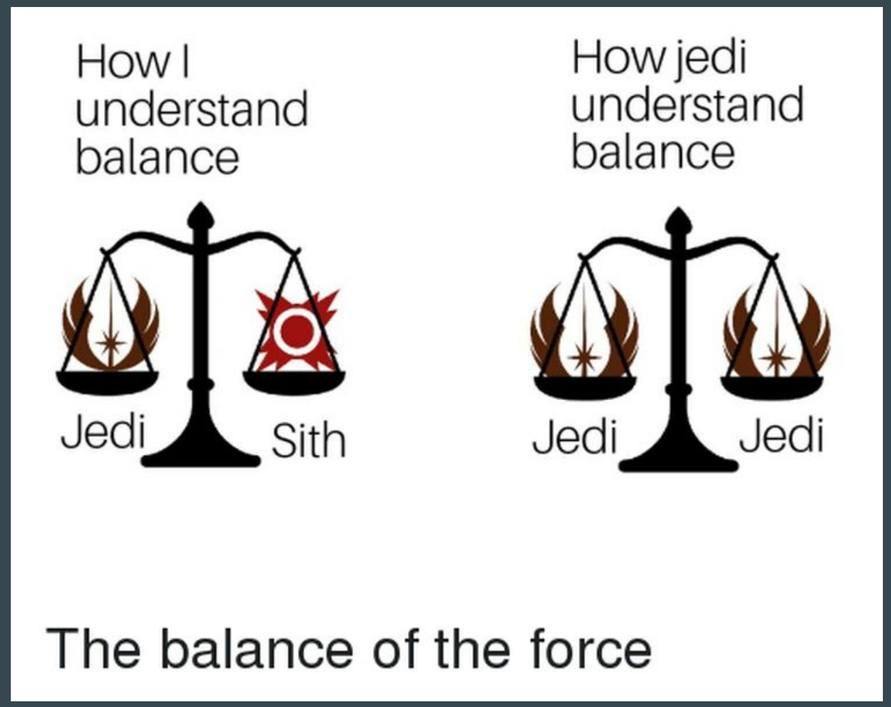
- Crackmes by NoraCodes
- The Rizin book
- Root Me
- crackmes.de (Press F)

Protect?

- Обфускаторы
- Протекторы
- Изменение архитектуры

Legal

Право на реверс. Как обратная разработка выглядит с юридической точки зрения



Questions? Ask @dukebarman

Image by Lucasfilm Ltd