The Poor Man's Obfuscator

ELF & Mach-O Tricks to Hinder Static Analysis

Romain Thomas

July, 2022

Pass The Salt

Introduction

About

- Security engineer at UL - La Ciotat
- Working on banking app certifications (EMVCo, VISA, ...)
- Author of LIEF: `https://lief.re`
- Enjoy Android, reverse engineering and, obfuscation.

The Challenges

1. Transform ELF & Mach-O binaries such as they look obfuscated

The Challenges

1. Transform ELF & Mach-O binaries such as they look obfuscated

2. Transformations **only** based on the executable formats

The Challenges

1. Transform ELF & Mach-O binaries such as they look obfuscated
2. Transformations **only** based on the executable formats
3. Must impact classical tools: IDA, BinaryNinja, Ghidra, Radare2 ...

The Challenges

1. Transform ELF & Mach-O binaries such as they look obfuscated
2. Transformations only based on the executable formats
3. Must impact classical tools: IDA, BinaryNinja, Ghidra, Radare2 ...
4. The modified binaries must still run after the transformations

Transformations Overview

- The transformations rely on LIEF (commit: `f8c711d`)
- The ELF and Mach-O `arm64` binaries used in this presentation come from the Mbed TLS test suite

Transformations Overview

Symbols

Sections

ELF / Mach-O

Symbols

Symbols

Exports: Random Names

```python
target = lief.parse("mbedtls_self_test.arm64.elf")

for function in target.functions:
    name = "".join(random.choice(ascii_letters) for i in range(20))
    target.add_exported_function(function.address, name)
```

```
Function name
start
ixjfmORQcjUlwkPkQGjj
ezFLyEHEncSjmFSdFRjM
MktrAESdDGzHAhFpfpLW
SsoGrpKNWPCTyzNFnIkR
lXXiMswzOgdozxYdhBAu
HnEyolibHdTkMkIubaXJ
ZxdXVNHfCYfBWLjyJNoK
OqeCqwtmDkPmnJtutBCk
VOfcDjRPNEUBTENIrzkA
LiRlxEEQDshSNnBxTmjG
ozGNdKdKRTKvZzvpCFSE
bkNkScONUVXvCcAXMjkh
llawFfmiPoFnFPSWdCkJ
zVkZwsDioGpRjytoBcyv       .tex          0198A8
eCmULlMDFXhDRVdJGGFB       .text   000000000001991C
NHOuuyKeiDgKffTOjEWU       .text   0000000000019990
drkgDBimhIqxzDpbPxGS       .text   0000000000019E74
XYkDKszVYoVzhqImnwac       .text   000000000001A358
tzCgAmYEGtLqNpsSeQtz       .text   000000000001A390
GAyIneuPBIjuZEDxMbTF       .text   000000000001A688
uXNjNeNXbbcLtBPevMZA       .text   000000000001ACDC
eKgbEFyJVBZlIcOtDsvd       .text   000000000001ADC8
pmKrSvWXJtKxQZGpjNPA       .text   000000000001AE6C
rFylGcEXTBLvXywPTnjq       .text   000000000001AF38
rZhvzzIBobbCwrELHxEA       .text   000000000001B8A8
xTKhbcqtNqTJydXKAlEV       .text   000000000001BF98
enYkeNefznGyjVhhbanv       .text   000000000001C19C
MIBTFybSKWucCCQRdunu       .text   000000000001C544
eCQpcPENsgNkXKmbpRZj       .text   000000000001C574
oMFpvkgdcovfnbjkzKOF       .text   000000000001C584
knYikcaEcNHWiHjUbuoj       .text   000000000001C860
XUGLNlbiZleQfqFrXhtV       .text   000000000001C94C
iNUzDftqxylrFiwSCJTQ       .text   000000000001CA18
FzceZqsbvuaTRTPtDEZg       .text   000000000001D43C
BTtKsOraFYrghxkJSEwN       .text   000000000001D528
oxzOoxnnhBeiHMKCimzS       .text   000000000001D560
zgupvXDShORQBhpIopjF       .text   000000000001D610
tggccOtQrtuioSiHRqOm       .text   000000000001D7BC
XmsaCXUUqhHhkGOdRVRc       .text   000000000001DA14
OUvyUkFXcKoHnHGEXIjN       .text   000000000001DBCC
VugDsZNVCMKYfpAveZzH       .text   000000000001DBDC
```

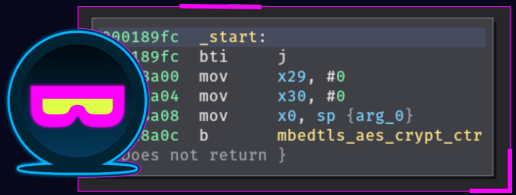
Symbols

Exports: Confusing Names

```
target  = lief.parse("mbedtls_self_test.arm64.elf")
nostrip = lief.parse("mbedtls_self_test.nostrip.arm64.elf")

symbols = [s.name for s in non_striped.symbols if s.startswith("mbedtls_")]

for function in target.functions:
    sym = random.choice(SYMBOLS)
    target.add_exported_function(function.address, sym)
```

Function name		
start		
mbedtls_aes_crypt_ctr		
mbedtls_ccm_encrypt_and_tag		
mbedtls_gcm_init		
mbedtls_chacha20_init		
mbedtls_rsa_private		
mbedtls_ctr_drbg_self_test		
mbedtls_ct_mpi_uint_cond_assign		
mbedtls_cmac_self_test		
mbedtls_des3_init		
mbedtls_sha256_info		
mbedtls_mpi_shift_r		
mbedtls_aria_crypt_ctr		
mbedtls_asn1_get_len		
mbedtls_internal_rlpend160_process	.text	000000000196A8
mbedtls_aes_init	.text	000000000199IC
mbedtls_ct_rsaes_pkcs1_v15_unpaddlng	.text	00000000000019990
mbedtls_sha512_update	.text	00000000000019E74
mbedtls_des3_set3key_enc	.text	000000000001A358
mbedtls_aria_crypt_ecb	.text	000000000001A390
mbedtls_cipher_setup	.text	000000000001A688
mbedtls_poly1305_self_test	.text	000000000001ACDC
mbedtls_camellia_crypt_ecb	.text	000000000001ADC8
mbedtls_mpi_fill_random	.text	000000000001AE6C
mbedtls_ccm_setkey	.text	000000000001AF38
mbedtls_ctr_drbg_reseed_internal	.text	000000000001BBA8
mbedtls_aria_crypt_cfb128	.text	000000000001BF9B
mbedtls_base64_self_test	.text	000000000001C19C
mbedtls_mpi_mul_int	.text	000000000001C544
mbedtls_des_free	.text	000000000001C574
mbedtls_sha512_self_test	.text	000000000001C584
mbedtls_nist_kw_init	.text	000000000001C860
mbedtls_ct_base64_enc_char	.text	000000000001C94C
mbedtls_md	.text	000000000001CA18
mbedtls_mpi_randon	.text	000000000001D43C
mbedtls_ecp_group_free	.text	000000000001D528
mbedtls_ecp_gen_keypair_base	.text	000000000001D560
mbedtls_chachapoly_init	.text	000000000001D610
mbedtls_mpi_lset	.text	000000000001D7BC
mbedtls_chachapoly_encrypt_and_tag	.text	000000000001DA14
mbedtls_cipher_cmac_finish	.text	000000000001DBCC
mbedtls ecjpake write round one	.text	000000000001DBDC

Exports: Confusing Names



Function name		
start		
mbedtls_aes_crypt_ctr		
mbedtls_ccm_encrypt_and_tag		
mbedtls_gcm_init		
mbedtls_chacha20_init		
mbedtls_rsa_private		
mbedtls_ctr_drbg_self_test		
mbedtls_ct_mpi_uint_cond_assign		
mbedtls_cmac_self_test		
mbedtls_des3_init		
mbedtls_sha256_info		
mbedtls_mpi_shift_r		
mbedtls_aria_crypt_ctr		
mbedtls_asn1_get_len		
mbedtls_internal_rlpend160_process	.text	00000000000199A8
mbedtls_aes_init	.text	00000000000199IC
mbedtls_ct_rsaes_pkcs1_v15_unpaddlng	.text	00000000000019990
mbedtls_sha512_update	.text	0000000000019E74
mbedtls_des3_set3key_enc	.text	00000000000001A358
mbedtls_aria_crypt_ecb	.text	00000000000001A390
mbedtls_cipher_setup	.text	00000000000001A688
mbedtls_poly1305_self_test	.text	00000000000001ACDC
mbedtls_camellia_crypt_ecb	.text	00000000000001ADC8
mbedtls_mpi_fill_random	.text	00000000000001AE6C
mbedtls_ccm_setkey	.text	00000000000001AF38
mbedtls_ctr_drbg_reseed_internal	.text	00000000000001BBA8
mbedtls_aria_crypt_cfb128	.text	00000000000001BF9B
mbedtls_base64_self_test	.text	00000000001C19C
mbedtls_mpi_mul_int	.text	00000000001C544
mbedtls_des_free	.text	00000000001C574
mbedtls_sha512_self_test	.text	00000000001C584
mbedtls_nist_kw_init	.text	00000000001C860
mbedtls_ct_base64_enc_char	.text	00000000001C94C
mbedtls_md	.text	00000000001CA18
mbedtls_mpi_randon	.text	00000000001D43C
mbedtls_ecp_group_free	.text	00000000000001D528
mbedtls_ecp_gen_keypair_base	.text	00000000000001D560
mbedtls_chachapoly_init	.text	00000000000001D610
mbedtls_mpi_lset	.text	00000000000001D7BC
mbedtls_chachapoly_encrypt_and_tag	.text	00000000000001DA14
mbedtls_cipher_cmac_finish	.text	00000000000001DBCC
mbedtls ecjpake write round one	.text	00000000000001DBDC

Symbols

Exports: libc symbols

```
target  = lief.parse("mbedtls_self_test.arm64.elf")
libc    = lief.parse("aarch64-linux-android/23/libc.so")

libc_symbols  = {s.name for s in libc.exported_symbols}
libc_symbols -= {s.name for s in target.imported_symbols}

for function in target.functions:
    sym = random.choice(libc_symbols)
    libc_symbols.remove(sym)

    export = target.add_exported_function(function.address, sym)

    export.binding    = lief.ELF.SYMBOL_BINDINGS.GNU_UNIQUE
    export.visibility = lief.ELF.SYMBOL_VISIBILITY.INTERNAL
```

Exports: libc symbols

Exports: libc symbols

Symbols


Create Exports
Fake Names
Fake Names from libc.so
Unalign Addresses

Exports: Unaligned Functions

```
address = function.address
address += random.randint(16, 32)
address -= address % 4

export = target.add_exported_function(address, sym)
```

The idea is to create exports with **unaligned** functions

Symbols

```
0001809c  mbedtls_aes_setkey_enc:
0001809c  stp     x29, x30, [sp, #-0x60]! {__saved_x29} {__saved_x30}
000180a0  stp     x28, x27, [sp, #0x10] {__saved_x28} {__saved_x27}
000180a4  stp     x26, x25, [sp, #0x20] {__saved_x26} {__saved_x25}
000180a8  stp     x24, x23, [sp, #0x30] {__saved_x24} {__saved_x23}
000180ac  stp     x22, x21, [sp, #0x40] {__saved_x22} {__saved_x21}
000180b0  stp     x20, x19, [sp, #0x50] {__saved_x20} {__saved_x19}
000180b4  mov     x29, sp {__saved_x29}
000180b8  sub     sp, sp, #0x840
000180bc  mrs     x3, tpidr_el0
000180c0  ldr     x9, [x3, #0x28]
000180c4  cmp     w2, #0x80
000180c8  stur    x9, [x29, #-0x10 {var_70}]
000180cc  b.eq    0x180f0
```

```
000180f0  mov     w9, #0xa
```

```
000180d0  cmp     w2, #0x100
000180d4  b.eq    0x180e8
```

```
000180e8  mov     w9, #0xe
000180ec  b       0x180f4
```

```
000180d8  cmp     w2, #0xc0
000180dc  b.ne    0x184dc
```

```
000184dc  mov     w0, #0xffffffe0  {0xffffffe0}
000184e0  b       0x185fc
```

```
000180e0  mov     w9, #0xc
000180e4  b       0x180f4
```

Symbols

```
0001909c  sub_1909c:
0001909c  stp     x29, x30, [sp, #-0x60]! {var_60} {var_58}
000190a0  stp     x28, x27, [sp, #0x10] {var_50} {var_48}
000190a4  stp     x26, x25, [sp, #0x20] {var_40} {var_38}
000190a8  stp     x24, x23, [sp, #0x30] {var_30} {var_28}
000190ac  stp     x22, x21, [sp, #0x40] {var_20}
000190b0  stp     x20, x19, [sp, #0x50] {var_1
000190b4  mov     x29, sp {var_60}  {mktemp}
{ Falls through into mktemp }
```

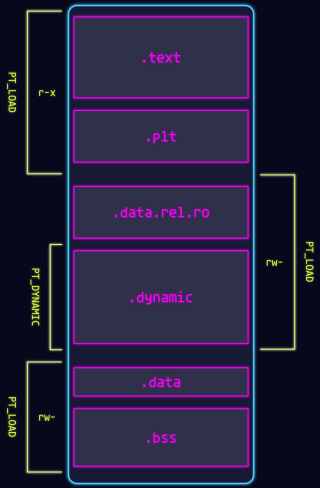
Symbols

```
[0x0001909c]> pdb
            ; XREFS: CALL 0x00019688  CALL 0x000198e4  CODE 0x00019904  CALL 0x00019958  CALL 0x0001b028  CALL 0x0001b110
            ; XREFS: CALL 0x0001b244  CALL 0x0001b424  CALL 0x0001b5a4  CALL 0x0001b734  CALL 0x0001b76c  CALL 0x0001b780
            ; XREFS: CODE 0x0002848c  CALL 0x0002a66c  CALL 0x0002a808  CALL 0x0002a9c4  CALL 0x0002ae04  CALL 0x0002b010
┌ 28: fcn.0001909c (int64_t arg1, int64_t arg2, int64_t arg3, int64_t arg_10h, int64_t arg_20h, int64_t arg_30h, int64_t
│ rg: 3 (vars 0, args 3)
│ bp: 0 (vars 0, args 0)
│ sp: 21 (vars 13, args 8)
│           0x0001909c      fd7bbaa9       stp x29, x30, [sp, -0x60]!
│           0x000190a0      fc6f01a9       stp x28, x27, [sp, 0x10]
│           0x000190a4      fa6702a9       stp x26, x25, [sp, 0x20]
│           0x000190a8      f85f03a9       stp x24, x23, [sp, 0x30]
│           0x000190ac      f65704a9       stp x22, x21, [sp, 0x40]
│           0x000190b0      f44f05a9       stp x20, x19, [sp, 0x50]
└           0x000190b4      fd030091       mov x29, sp
[0x0001909c]>
```

Sections

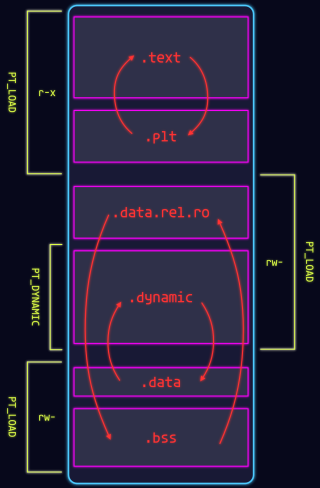
Sections

Parsing an ELF binary from sections is error-prone.



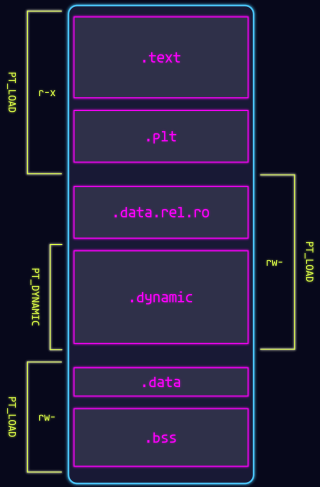
Sections

```
SWAP_LIST = [
    (".rela.dyn", ".data.rel.ro"),
    (".got",      ".data"),
    (".plt",      ".text"),

    (".preinit_array", ".bss"),
]

for (lhs_name, rhs_name) in SWAP_LIST:
    # ...
    lhs.offset          = rhs.offset
    lhs.size            = rhs.size
    lhs.name            = rhs.name
    lhs.type            = rhs.type
    lhs.virtual_address = rhs.virtual_address
    # ...
```



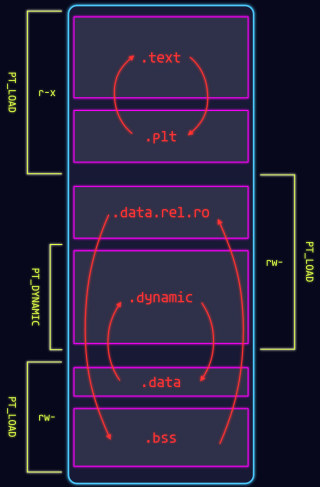
Sections

```
.text:000000000003F4A4    LDP     W16, W2, [X9,#-0x1C]
.text:000000000003F4A8    LDR     W17, [X10,#0x4C]
.text:000000000003F4AC    EOR     W13, W13, W12,ROR#11
.text:000000000003F4B0    ORR     W14, W15, W14
.text:000000000003F4B4    EOR     W12, W13, W12,ROR#25
.text:000000000003F4B8    LDP     W15, W13, [X10,#-0x2C]
.text:000000000003F4BC    LDUR    W1, [X10,#-0]
.text:000000000003F4C0    ADD     W12, W12, W4
.text:000000000003F4C4    ADD     W12, W12, W16
.text:000000000003F4C8    EXTR    W16, W17, W17, #0x11
.text:000000000003F4CC    ADD     W12, W12, W14
.text:000000000003F4D0    EXTR    W14, W13, W13, #7
.text:000000000003F4D4    EOR     W16, W16, W17,ROR#19
.text:000000000003F4D8    EOR     W14, W14, W13,ROR#18
.text:000000000003F4DC    EOR     W16, W16, W17,LSR#10
.text:000000000003F4E0    EOR     W13, W14, W13,LSR#3
.text:000000000003F4E4    ADD     W14, W16, W1
.text:000000000003F4E8    ADD     W14, W14, W15
.text:000000000003F4EC    ADD     W13, W14, W13
.text:000000000003F4F0    STR     W13, [X10,#0x14]
.text:000000000003F4F4    ADD     W12, W12, W13
.text:000000000003F4F8    LDR     W13, [SP,#0x130+var_28]
.text:000000000003F4FC    LDR     W14, [SP,#0x130+var_24]
.text:000000000003F500    LDR     W15, [SP,#0x130+var_20]
.text:000000000003F504    LDR     W16, [SP,#0x130+var_1C]
.text:000000000003F508    LDR     W17, [SP,#0x130+var_10]
.text:000000000003F50C    ORR     W1, W14, W13
.text:000000000003F510    AND     W15, W15, W1
.text:000000000003F514    EXTR    W1, W13, W13, #2
.text:000000000003F518    AND     W14, W14, W13
.text:000000000003F51C    EOR     W1, W1, W13,ROR#13
.text:000000000003F520    EOR     W13, W1, W13,ROR#22
.text:000000000003F524    LDR     W1, [SP,#0x130+var_14]
.text:000000000003F528    ORR     W14, W15, W14
.text:000000000003F52C    LDR     W15, [SP,#0x130+var_18]
.text:000000000003F530    ADD     W16, W16, W12
.text:000000000003F534    BIC     W1, W1, W16
.text:000000000003F538    ADD     W13, W14, W13
.text:000000000003F53C    AND     W15, W15, W16
.text:000000000003F540    ORR     W15, W15, W1
.text:000000000003F544    EXTR    W1, W16, W16, #6
.text:000000000003F548    EOR     W1, W1, W16,ROR#11
.text:000000000003F54C    EOR     W14, W1, W16,ROR#25
.text:000000000003F550    ADD     W12, W13, W12
.text:000000000003F554    STR     W16, [SP,#0x130+var_1C]
```



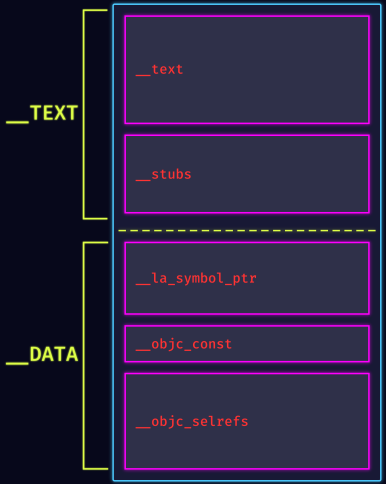
Sections

```
LOAD:0000000000003F4A0                 AND             W15, W16, W12
LOAD:0000000000003F4A4                 LDP             W16, W2, [X9,#-0x1C]
LOAD:0000000000003F4A8                 LDR             W17, [X10,#0x4C]
LOAD:0000000000003F4AC                 EOR             W13, W13, W12,ROR#11
LOAD:0000000000003F4B0                 ORR             W14, W15, W14
LOAD:0000000000003F4B4                 EOR             W12, W13, W12,ROR#25
LOAD:0000000000003F4B8                 LDP             W15, W13, [X10,#-0x2C]
LOAD:0000000000003F4BC                 LDUR            W1, [X10,#-8]
LOAD:0000000000003F4C0                 ADD             W12, W12, W4
LOAD:0000000000003F4C4                 ADD             W12, W12, W16
LOAD:0000000000003F4C8                 EXTR            W16, W17, W17, #0x11
LOAD:0000000000003F4CC                 ADD             W12, W12, W14
LOAD:0000000000003F4D0                 EXTR            W14, W13, W13, #7
LOAD:0000000000003F4D4                 EOR             W16, W16, W17,ROR#19
LOAD:0000000000003F4D8                 EOR             W14, W14, W13,ROR#18
LOAD:0000000000003F4DC                 EOR             W16, W16, W17,LSR#10
LOAD:0000000000003F4E0                 EOR             W13, W14, W13,LSR#3
LOAD:0000000000003F4E4                 ADD             W14, W16, W1
LOAD:0000000000003F4E8                 ADD             W14, W14, W15
LOAD:0000000000003F4EC                 ADD             W13, W14, W13
LOAD:0000000000003F4F0                 STR             W13, [X10,#0x14]
LOAD:0000000000003F4F4 ; ---------------------------------------------------------------------------
LOAD:0000000000003F4F4 dword_3F4F4     DCD 0x B0D018C          ; DATA XREF: LOAD:0000
LOAD:0000000000003F4F8 ; ---------------------------------------------------------------------------
LOAD:0000000000003F4FC                 LDR             W13, [SP,#0x108]
LOAD:0000000000003F500                 LDR             W14, [SP,#0x10C]
LOAD:0000000000003F504                 LDR             W15, [SP,#0x110]
LOAD:0000000000003F508                 LDR             W16, [SP,#0x114]
LOAD:0000000000003F50C                 LDR             W17, [SP,#0x120]
LOAD:0000000000003F510                 ORR             W1, W14, W13
LOAD:0000000000003F514                 AND             W15, W15, W1
LOAD:0000000000003F518                 EXTR            W1, W13, W13, #2
LOAD:0000000000003F51C                 AND             W14, W14, W13
LOAD:0000000000003F520                 EOR             W1, W1, W13,ROR#13
LOAD:0000000000003F524                 EOR             W13, W1, W13,ROR#22
LOAD:0000000000003F528                 LDR             W1, [SP,#0x11C]
LOAD:0000000000003F52C                 ORR             W14, W15, W14
LOAD:0000000000003F530                 LDR             W15, [SP,#0x118]
LOAD:0000000000003F534                 ADD             W16, W16, W1
LOAD:0000000000003F538                 BIC             W1, W1, W16
LOAD:0000000000003F53C                 ADD             W13, W14, W13
LOAD:0000000000003F540                 AND             W15, W15, W16
LOAD:0000000000003F544                 ORR             W15, W15, W1
LOAD:0000000000003F548                 EXTR            W1, W16, W16, #6
LOAD:0000000000003F54C                 EOR             W1, W1, W16,ROR#11
```



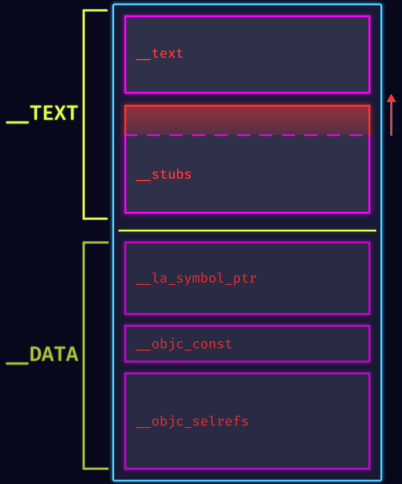
Sections

The Mach-O format and dyld enforce a stricter layout for sections.

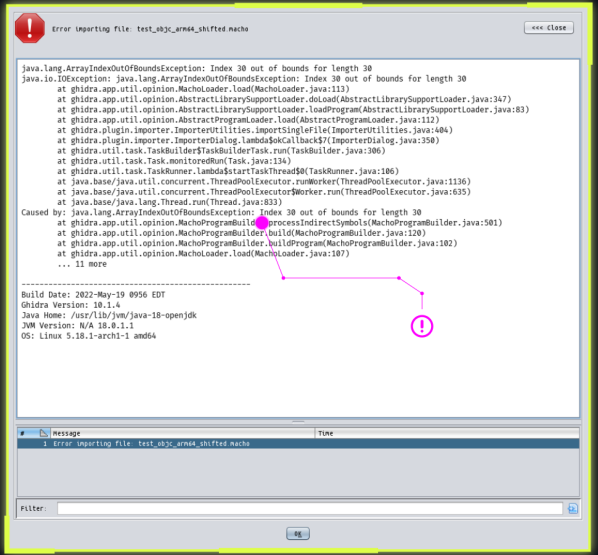


Sections

```
__text  = target.get_section("__text")
__stubs = target.get_section("__stubs")

DELTA = 0x100

__text.size             -= DELTA

__stubs.offset          -= DELTA
__stubs.virtual_address -= DELTA
__stubs.size            += DELTA
```



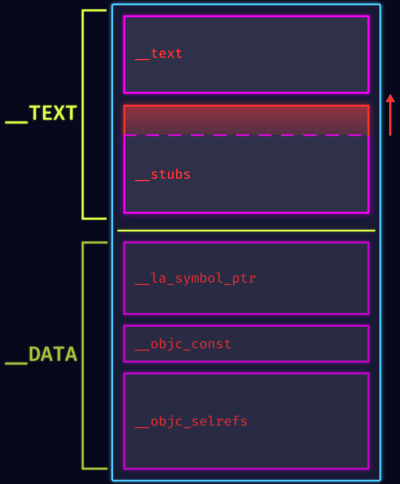
Sections


Error importing file: test_objc_arm64_shifted.macho
<<< Close
java.lang.ArrayIndexOutOfBoundsException: Index 30 out of bounds for length 30
java.io.IOException: java.lang.ArrayIndexOutOfBoundsException: Index 30 out of bounds for length 30
at ghidra.app.util.opinion.MachoLoader.load(MachoLoader.java:113)
at ghidra.app.util.opinion.AbstractLibrarySupportLoader.doLoad(AbstractLibrarySupportLoader.java:347)
at ghidra.app.util.opinion.AbstractLibrarySupportLoader.loadProgram(AbstractLibrarySupportLoader.java:83)
at ghidra.app.util.opinion.AbstractProgramLoader.load(AbstractProgramLoader.java:112)
at ghidra.plugin.importer.ImporterUtilities.importSingleFile(ImporterUtilities.java:404)
at ghidra.plugin.importer.ImporterDialog.lambda$okCallback$7(ImporterDialog.java:350)
at ghidra.util.task.TaskBuilder$TaskBuilderTask.run(TaskBuilder.java:306)
at ghidra.util.task.Task.monitoredRun(Task.java:134)
at ghidra.util.task.TaskRunner.lambda$startTaskThread$0(TaskRunner.java:106)
at java.base/java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1136)
at java.base/java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:635)
at java.base/java.lang.Thread.run(Thread.java:833)
Caused by: java.lang.ArrayIndexOutOfBoundsException: Index 30 out of bounds for length 30
at ghidra.app.util.opinion.MachoProgramBuilder.processIndirectSymbols(MachoProgramBuilder.java:501)
at ghidra.app.util.opinion.MachoProgramBuilder.build(MachoProgramBuilder.java:120)
at ghidra.app.util.opinion.MachoProgramBuilder.buildProgram(MachoProgramBuilder.java:102)
at ghidra.app.util.opinion.MachoLoader.load(MachoLoader.java:107)
... 11 more
Build Date: 2022-May-19 0956 EDT
Ghidra Version: 10.1.4
Java Home: /usr/lib/jvm/java-18-openjdk
JVM Version: N/A 18.0.1.1
OS: Linux 5.18.1-arch1-1 amd64
#
Message
Time
1 Error importing file: test_objc_arm64_shifted.macho
Filter:
OK


__TEXT
__text
__stubs
__DATA
__la_symbol_ptr
__objc_const
__objc_selrefs

Sections

```
; int __cdecl main(int argc, const char **argv, const char **envp)
EXPORT _main
_main

var_10= -0x10

SUB             SP, SP, #0x40
STP             X29, X30, [SP,#0x40+var_10]
; End of function _main
```



Sections

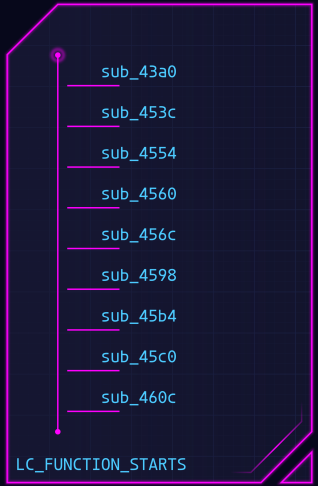
```
__stubs:0000000100003784 ; ═══════════ S U B R O U T I N E ═══════════
__stubs:0000000100003784
__stubs:0000000100003784 ; Attributes: noreturn
__stubs:0000000100003784
__stubs:0000000100003784 ; void NSLog(NSString *format, ...)
__stubs:0000000100003784 _NSLog
__stubs:0000000100003784
__stubs:0000000100003784 arg_8           =  8
__stubs:0000000100003784 arg_20          =  0x20
__stubs:0000000100003784
__stubs:0000000100003784                 LDR             X0, [SP,#arg_8]
__stubs:00000001000037B8                 LDP             X29, X30, [SP,#arg_20]
__stubs:00000001000037BC                 ADD             SP, SP, #0x10 ; '0'
__stubs:00000001000037C0
__stubs:00000001000037C0 ; void __cdecl _Unwind_Resume(_Unwind_Exception *exception_object)
__stubs:00000001000037C0 __Unwind_Resume
__stubs:00000001000037C0                 RET
__stubs:00000001000037C0 ; End of function _NSLog
__stubs:00000001000037C0
__stubs:00000001000037C4
__stubs:00000001000037C4 ; ═══════════ S U B R O U T I N E ═══════════
__stubs:00000001000037C4
__stubs:00000001000037C4 ; Attributes: noreturn bp-based frame
__stubs:00000001000037C4
__stubs:00000001000037C4 ; std::allocator<char>::allocator(void)
__stubs:00000001000037C4 __ZNSt3__19allocatorIcEC2Ev_0           ; CODE XREF: std::__compressed_pair_ele
__stubs:00000001000037C4
__stubs:00000001000037C4 var_10          = -0x10
__stubs:00000001000037C4 var_8           = -8
__stubs:00000001000037C4 var_s0          =  0
__stubs:00000001000037C4
__stubs:00000001000037C4                 SUB             SP, SP, #0x20
__stubs:00000001000037C8                 STP             X29, X30, [SP,#0x10+var_s0]
__stubs:00000001000037CC
__stubs:00000001000037CC ; std::char_traits<char>::length(char const*)
__stubs:00000001000037CC ) ___ZNSt3__11Ichar_traitsIcE6lengthEPKc
__stubs:00000001000037CC                 ADD             X29, SP, #0x10
__stubs:00000001000037D0                 STR             X0, [SP,#0x10+var_8]
__stubs:00000001000037D4                 LDR             X0, [SP,#0x10+var_8]
```

Specific Transformations

Mach-O: LC_FUNCTION_STARTS

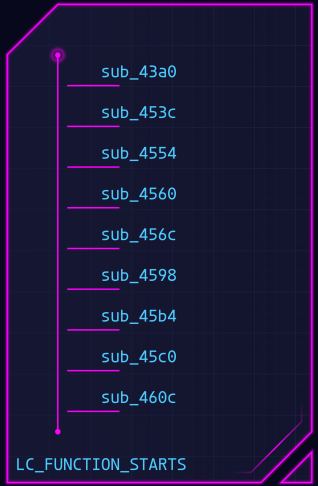
The LC_FUNCTION_STARTS is a Mach-O command that embeds a list of functions.

Similarly to unaligned exports, we can unalign these functions

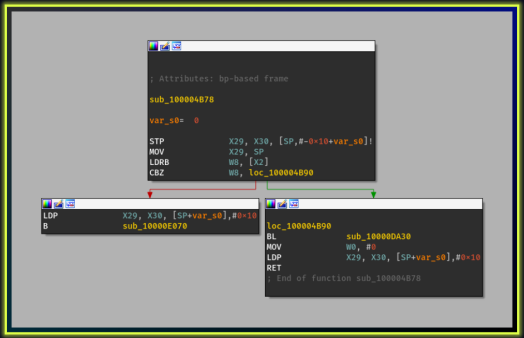


Mach-O: LC_FUNCTION_STARTS

```
functions = [addr for addr in LC_FUNCTION_STARTS.functions]

for idx, _ in enumerate(functions):
    if idx % 2 == 0:
        functions[idx] += 4 * 7
    else:
        functions[idx] -= 4 * 7

LC_FUNCTION_STARTS.functions = functions
```



Mach-O: LC_FUNCTION_STARTS



Mach-O: LC_FUNCTION_STARTS

```
; Attributes: bp-based frame

sub_100004B78

var_s0=  0

STP             X29, X30, [SP,#-0x10+var_s0]!
MOV             X29, SP
LDRB            W8, [X2]
CBZ             W8, loc_100004B90
```

```
LDP             X29, X30, [SP+var_s0],#0x10
B               sub_10000E070
```

```
loc_100004B90
BL              sub_10000DA30
MOV             W0, #0
LDP             X29, X30, [SP+var_s0],#0x10
RET
; End of function sub_100004B78
```

```
; Attributes: bp-based frame

sub_100004B78

var_s0=  0

STP             X29, X30, [SP,#-0x10+var_s0]!
MOV             X29, SP
LDRB            W8, [X2]
; End of function sub_100004B78
```

Mach-O: LC_FUNCTION_STARTS

```
__text:0000000100004B1C sub_100004B1C
__text:0000000100004B1C
__text:0000000100004B1C arg_10          =  0x10
__text:0000000100004B1C
__text:0000000100004B1C                 LDR             X8, =__stderrp
__text:0000000100004B20                 LDR             X8, [X8]
__text:0000000100004B24                 ADR             X8, aObufNull ; "obuf != NULL"
__text:0000000100004B28                 NOP
__text:0000000100004B2C                 STR             X8, [SP,#arg_10]
__text:0000000100004B30                 MOV             W8, #0xCE
__text:0000000100004B34                 B               loc_1000004AF8
__text:0000000100004B34 ; End of function sub_100004B1C
__text:0000000100004B34
__text:0000000100004B38 ; ---------------------------------------------------------------------------
__text:0000000100004B38                 CMP             W2, W3
__text:0000000100004B3C                 B.NE            loc_100004B70
__text:0000000100004B40                 CBZ             W2, loc_100004B68
__text:0000000100004B44                 MOV             W8, W2
__text:0000000100004B48
__text:0000000100004B48 loc_100004B48                           ; CODE XREF: __text:0000000100004B44↓j
__text:0000000100004B48                 LDRB            W9, [X0]
__text:0000000100004B4C                 LDRB            W10, [X1]
__text:0000000100004B50                 CMP             W9, W10
__text:0000000100004B54                 B.NE            loc_100004B70
__text:0000000100004B58                 ADD             X0, X0, #1
__text:0000000100004B5C                 ADD             X1, X1, #1
__text:0000000100004B60                 SUBS            X8, X8, #1
__text:0000000100004B64                 B.NE            loc_100004B48
__text:0000000100004B68
__text:0000000100004B68 loc_100004B68                           ; CODE XREF: __text:0000000100004B40↑j
__text:0000000100004B68                 MOV             W0, #0
__text:0000000100004B6C                 RET
__text:0000000100004B70 ; ---------------------------------------------------------------------------
__text:0000000100004B70
__text:0000000100004B70 loc_100004B70                           ; CODE XREF: __text:0000000100004B3C↑j
__text:0000000100004B70                                         ; __text:0000000100004B54↑j
__text:0000000100004B70                 MOV             W0, #0xFFFFFFFF
__text:0000000100004B74                 RET
__text:0000000100004B78
__text:0000000100004B78 ; =============== S U B R O U T I N E =======================================
__text:0000000100004B78
__text:0000000100004B78 ; Attributes: bp-based frame
__text:0000000100004B78
__text:0000000100004B78 sub_100004B78
__text:0000000100004B78
__text:0000000100004B78 var_s0          =  0
__text:0000000100004B78
__text:0000000100004B78                 STP             X29, X30, [SP,#-0x10+var_s0]!
__text:0000000100004B7C                 MOV             X29, SP
__text:0000000100004B80                 LDRB            W8, [X2]
__text:0000000100004B80 ; End of function sub_100004B78
__text:0000000100004B80
```

Mach-O: LC_FUNCTION_STARTS

```
100004b38  5f00036b  cmp     w2, w3
100004b3c  a1010054  b.ne    0x100004b70

100004b40  42010034  cbz     w2, 0x100004b68

100004b44  e803022a  mov     w8, w2

100004b48  09004039  ldrb    w9, [x0]
100004b4c  2a004039  ldrb    w10, [x1]
100004b50  3f010a6b  cmp     w9, w10
100004b54  e1000054  b.ne    0x100004b70

100004b58  00040091  add     x0, x0, #0x1
100004b5c  21040091  add     x1, x1, #0x1
100004b60  080500f1  subs    x8, x8, #0x1
100004b64  21ffff54  b.ne    0x100004b48

100004b68  00008052  mov     w0, #0
100004b6c  c0035fd6  ret

100004b70  00008012  mov     w0, #0xffffffff  {0xffffffff}
100004b74  c0035fd6  ret

100004b78  int64_t sub_100004b78(int32_t* arg1, int32_t arg2, char* arg3)

100004b78  fd7bbfa9  stp     x29, x30, [sp, #-0x10]! {__saved_x29} {__saved_x30}
100004b7c  fd030091  mov     x29, sp {__saved_x29}
100004b80  48004039  ldrb    w8, [x2]
100004b84  68000034  cbz     w8, 0x100004b90

100004b88  fd7bc1a8  ldp     x29, x30, [sp], #0x10 {__saved_x29} {__saved_x30}
100004b8c  39250014  b       sub_10000e070

100004b90  a8230094  bl      sub_10000da30
100004b94  00008052  mov     w0, #0
100004b98  fd7bc1a8  ldp     x29, x30, [sp], #0x10 {__saved_x29} {__saved_x30}
100004b9c  c0035fd6  ret

100004ba0  int64_t sub_100004ba0(int32_t arg1)

100004ba0  8002f837  tbnz    w0, #0x1f, 0x100004bf0

100004ba4  480300f0  adrp    x8, 0x10006f000
100004ba8  092543f9  ldr     x9, [x8, #0x648]  {data_10006f648}
100004bac  2a553510  adr     x10, 0x10006f650
100004bb0  1f2003d5  nop
100004bb4  890100b4  cbz     x9, 0x100004be4
```

Mach-O: LC_FUNCTION_STARTS

```
100004b38  5f00036b  cmp     w2, w3
100004b3c  a1010054  b.ne    0x100004b70

100004b40  42010034  cbz     w2, 0x100004b68

100004b44  e803022a  mov     w8, w2

100004b48  09004039  ldrb    w9, [x0]
100004b4c  2a004039  ldrb    w10, [x1]
100004b50  3f010a6b  cmp     w9, w10
100004b54  e1000054  b.ne    0x100004b70

100004b58  00040091  add     x0, x0, #0x1
100004b5c  21040091  add     x1, x1, #0x1
100004b60  080500f1  subs    x8, x8, #0x1
100004b64  21ffff54  b.ne    0x100004b48

100004b68  00008052  mov     w0, #0
100004b6c  c0035fd6  ret

100004b70  00008012  mov     w0, #0xffffffff  {0xffffffff}
100004b74  c0035fd6  ret
```

```
100004b78  int64_t sub_100004b78(int32_t* arg1, int32_t arg2, char* arg3)

100004b78  fd7bbfa9  stp     x29, x30, [sp, #-0x10]! {__saved_x29} {__saved_x30}
100004b7c  fd030091  mov     x29, sp {__saved_x29}
100004b80  48004039  ldrb    w8, [x2]
100004b84  68000034  cbz     w8, 0x100004b90

100004b88  fd7bc1a8  ldp     x29, x30, [sp], #0x10 {__saved_x29} {__saved_x30}
100004b8c  39250014  b       sub_10000e070

100004b90  a8230094  bl      sub_10000da30
100004b94  00008052  mov     w0, #0
100004b98  fd7bc1a8  ldp     x29, x30, [sp], #0x10 {__saved_x29} {__saved_x30}
100004b9c  c0035fd6  ret
```

```
100004ba0  int64_t sub_100004ba0(int32_t arg1)

100004ba0  8002f837  tbnz    w0, #0x1f, 0x100004bf0

100004ba4  480300f0  adrp    x8, 0x10006f000
100004ba8  092543f9  ldr     x9, [x8, #0x648]  {data_10006f648}
100004bac  2a553510  adr     x10, 0x10006f650
100004bb0  1f2003d5  nop
100004bb4  890100b4  cbz     x9, 0x100004be4

100004bb8  0b0080d2  mov     x11, #0

100004bbc  4c796bb8  ldr     w12, [x10, x11, lsl #0x2]
100004bc0  9f01006b  cmp     w12, w0
100004bc4  60010054  b.eq    0x100004bf0
```

```
100004b50  3f010a6b  cmp     w9, w10
100004b54  e1000054  b.ne    0x100004b70

100004b58  00040091  add     x0, x0, #0x1
100004b5c  21040091  add     x1, x1, #0x1
100004b60  080500f1  subs    x8, x8, #0x1
100004b64  21ffff54  b.ne    0x100004b48

100004b68  00008052  mov     w0, #0
100004b6c  c0035fd6  ret

100004b70  00008012  mov     w0, #0xffffffff  {0xffffffff}
100004b74  c0035fd6  ret
```

```
100004b78
100004b80  48 00 40 39
```

```
100004b84  int64_t sub_100004b84(int32_t* arg1, int32_t arg2, char* arg3, int32_t

100004b84  68000034  cbz     w8, 0x100004b90

100004b88  fd7bc1a8  ldp     x29, x30, [sp], #0x10 {arg5} {arg6}
100004b8c  39250014  b       sub_10000e070

100004b90  a8230094  bl      sub_10000da30  {sub_100004b94}
{ Falls through into sub_100004b94 }
```

```
100004b94  int64_t sub_100004b94(int64_t arg1)

100004b94  00008052  mov     w0, #0
100004b98  fd7bc1a8  ldp     x29, x30, [sp], #0x10 {arg1} {arg_8}
100004b9c  c0035fd6  ret
```

```
100004ba0  80 02 f8 37
```

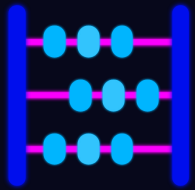
```
100004ba4  int64_t sub_100004ba4(int32_t arg1)

100004ba4  480300f0  adrp    x8, 0x10006f000
100004ba8  092543f9  ldr     x9, [x8, #0x648]  {data_10006f648}
100004bac  2a553510  adr     x10, 0x10006f650
100004bb0  1f2003d5  nop
100004bb4  890100b4  cbz     x9, 0x100004be4

100004bb8  0b0080d2  mov     x11, #0

100004bbc  4c796bb8  ldr     w12, [x10, x11, lsl #0x2]
100004bc0  9f01006b  cmp     w12, w0
100004bc4  60010054  b.eq    0x100004bf0
```

Mach-O: LC_FUNCTION_STARTS

```
[0x100004b50]> pd 20
        0x100004b50      3f010a6b       cmp w9, w10
        0x100004b54      e1000054       b.ne 0x100004b70
        0x100004b58      0104a091       add x0, x0, 1
        0x100004b5c      21040091       add x1, x1, 1
        0x100004b60      0805a0f1       subs x8, x8, 1
        0x100004b64      21ffff54       b.ne 0x100004b48
        0x100004b68      00008052       mov w0, 0
        0x100004b6c      c0035fd6       ret
        0x100004b70      00008012       mov w0, -1
        0x100004b74      c0035fd6       ret
┌ 12: fcn.100004b70 (int64_t arg1, int64_t arg2, int64_t arg3);
│          ; arg int64_t arg1 @ x0
│          ; arg int64_t arg2 @ x1
│          ; arg int64_t arg3 @ x2
│          ; var int64_t var_10h @ sp+0x0
│          ; var int64_t var_10h_2 @ sp+0x8
│          0x100004b78      fd7bbfa9       stp x29, x30, [var_10h]!
│          0x100004b7c      fd030091       mov x29, sp
└          0x100004b80      48004039       ldrb w8, [x2]              ; 0xda ; 218 ; arg3
┌ 16: sym.func.100004b84 (int64_t arg1, int64_t arg2, int64_t arg3);
│          ; arg int64_t arg1 @ x0
│          ; arg int64_t arg2 @ x1
│          ; arg int64_t arg3 @ x2
│          ; var int64_t var_10h @ sp+0x60
│          ; var int64_t var_20h @ sp+0x70
│          ; var int64_t var_30h @ sp+0x80
│          ; var int64_t var_40h @ sp+0x90
│       ┌< 0x100004b84      68000034       cbz w8, 0x100004b90
│       │  0x100004b88      fd7bc1a8       ldp x29, x30, [sp], 0x10
│       │< 0x100004b8c      39250014       b fcn.10000e070
└       └> 0x100004b90      a8230094       bl fcn.10000da30
┌ 12: sym.func.100004b94 ();
│          0x100004b94      00008052       mov w0, 0
│          0x100004b98      fd7bc1a8       ldp x29, x30, [sp], 0x10
└          0x100004b9c      c0035fd6       ret
[0x100004b50]>
(1) romain1:5.7.0*                                                    0.42 0.58 0.34 06:22
```

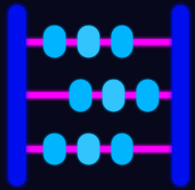
ELF: .dynsym

Counting the number of dynamic symbols in an ELF binary is somehow complicated ...



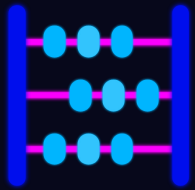
ELF: .dynsym

- Easy & Dirty: `.dynsym` section
- Harder & Reliable: DT_GNU_HASH / DT_HASH



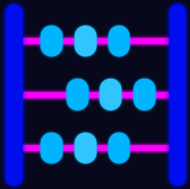
ELF: .dynsym

```
dynsym = target.get_section(".dynsym").as_frame()

sizeof = dynsym.entry_size
osize  = dynsym.size
nsyms  = osize / sizeof

dynsym.size = sizeof * min(3, nsyms)
```



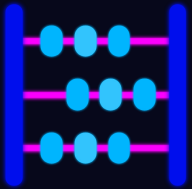
ELF: .dynsym

```
.plt:00000000000410A0 ; int __fastcall __cxa_atexit(void (__fastcall *lpfunc)(void *), void *obj, void *lpdso_handle)
.plt:00000000000410A0 .__cxa_atexit                           ; CODE XREF: sub_17A68+1C↑j
.plt:00000000000410A0                 ADRP            X16, #off_48880@PAGE
.plt:00000000000410A4                 LDR             X17, [X16,#off_48880@PAGEOFF]
.plt:00000000000410A8                 ADD             X16, X16, #off_48880@PAGEOFF
.plt:00000000000410AC                 BR              X17
.plt:00000000000410AC ; End of function .__cxa_atexit
.plt:00000000000410AC
.plt:00000000000410B0
.plt:00000000000410B0 ; ═══════════════ S U B R O U T I N E ═══════════════════════════
.plt:00000000000410B0
.plt:00000000000410B0 ; Attributes: thunk
.plt:00000000000410B0
.plt:00000000000410B0 ; int puts(const char *s)
.plt:00000000000410B0 .puts                                   ; CODE XREF: sub_17A8C+14↑p
.plt:00000000000410B0                                         ; sub_17A8C+20↑p  ...
.plt:00000000000410B0                 ADRP            X16, #off_48888@PAGE
.plt:00000000000410B4                 LDR             X17, [X16,#off_48888@PAGEOFF]
.plt:00000000000410B8                 ADD             X16, X16, #off_48888@PAGEOFF
.plt:00000000000410BC                 BR              X17
.plt:00000000000410BC ; End of function .puts
.plt:00000000000410BC
.plt:00000000000410C0
.plt:00000000000410C0 ; ═══════════════ S U B R O U T I N E ═══════════════════════════
.plt:00000000000410C0
.plt:00000000000410C0 ; Attributes: thunk
.plt:00000000000410C0
.plt:00000000000410C0 ; int printf(const char *format, ...)
.plt:00000000000410C0 .printf                                 ; CODE XREF: sub_17A8C+34↑p
.plt:00000000000410C0                                         ; sub_17AD8+32C↑p  ...
.plt:00000000000410C0                 ADRP            X16, #off_48890@PAGE
.plt:00000000000410C4                 LDR             X17, [X16,#off_48890@PAGEOFF]
.plt:00000000000410C8                 ADD             X16, X16, #off_48890@PAGEOFF
.plt:00000000000410CC                 BR              X17
.plt:00000000000410CC ; End of function .printf
.plt:00000000000410CC
```



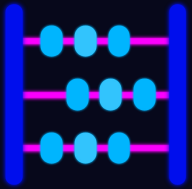
ELF: .dynsym

```
.plt:00000000000410A0 ; int __fastcall __cxa_atexit(void (__fastcall *lpfunc)(void *), void *obj, void *lpdso_handle)
.plt:00000000000410A0 .__cxa_atexit                    ; CODE XREF: sub_17A68+1C↑j
.plt:00000000000410A0                 ADRP            X16, #off_48880@PAGE
.plt:00000000000410A4                 LDR             X17, [X16,#off_48880@PAGEOFF]
.plt:00000000000410A8                 ADD             X16, X16, #off_48880@PAGEOFF
.plt:00000000000410AC                 BR              X17
.plt:00000000000410AC ; End of function .__cxa_atexit
.plt:00000000000410AC
.plt:00000000000410B0
.plt:00000000000410B0 ; ═══════════════ S U B R O U T I N E ═══════════════════════════
.plt:00000000000410B0
.plt:00000000000410B0 ; Attributes: thunk
.plt:00000000000410B0
.plt:00000000000410B0 sub_410B0                        ; CODE XREF: sub_17A8C+14↑p
.plt:00000000000410B0                                  ; sub_17A8C+20↑p  ...
.plt:00000000000410B0                 ADRP            X16, #qword_48888@PAGE
.plt:00000000000410B4                 LDR             X17, [X16,#qword_48888@PAGEOFF]
.plt:00000000000410B8                 ADD             X16, X16, #qword_48888@PAGEOFF
.plt:00000000000410BC                 BR              X17
.plt:00000000000410BC ; End of function sub_410B0
.plt:00000000000410BC
.plt:00000000000410C0
.plt:00000000000410C0 ; ═══════════════ S U B R O U T I N E ═══════════════════════════
.plt:00000000000410C0
.plt:00000000000410C0 ; Attributes: thunk
.plt:00000000000410C0
.plt:00000000000410C0 sub_410C0                        ; CODE XREF: sub_17A8C+34↑p
.plt:00000000000410C0                                  ; sub_17AD6+32C↑p  ...
.plt:00000000000410C0                 ADRP            X16, #qword_48890@PAGE
.plt:00000000000410C4                 LDR             X17, [X16,#qword_48890@PAGEOFF]
.plt:00000000000410C8                 ADD             X16, X16, #qword_48890@PAGEOFF
.plt:00000000000410CC                 BR              X17
.plt:00000000000410CC ; End of function sub_410C0
.plt:00000000000410CC
.plt:00000000000410D0
.plt:00000000000410D0 ; ═══════════════ S U B R O U T I N E ═══════════════════════════
```



ELF: .dynsym

```
000410c0 30 00 00 f0     adrp        x16,0xA0000
000410c4 11 4a 4a f9     ldr         x17,[x16, #0x890]=>PTR_00048090                    = 00041070
000410c8 10 42 22 91     add         x16,x16,#0x890
000410cc 20 02 1f d6     br          x17=>SUB_ffffffffffffa1070
                         -- Flow Override: CALL_RETURN (COMPUTED_CALL_TERMINATOR)

                         **************************************************************
                         *                          THUNK FUNCTION                    *
                         **************************************************************
                         thunk undefined thunk_EXT_FUN_00041070()
                           Thunked-Function: <EXTERNAL>::EXT_FUN_000...
     undefined           w0:1                <RETURN>
                         <EXTERNAL>::thunk_EXT_FUN_00041070                 XREF[86]:    FUN_00017a8c:00017ac0(c),
                                                                                         FUN_00017ad8:00017e04(c),
                                                                                         FUN_00017ad8:00017ed8(c),
                                                                                         FUN_00017ad8:00017f00(c),
                                                                                         FUN_00017a8c:00017f18(c),
                                                                                         FUN_00019f38:00019ffc(c),
                                                                                         FUN_00019f38:0001a0dc(c),
                                                                                         FUN_00019f38:0001a20c(c),
                                                                                         FUN_00019f38:0001a3ec(c),
                                                                                         FUN_00019f38:0001a568(c),
                                                                                         FUN_00019f38:0001a704(c),
                                                                                         FUN_0001ba18:0001ba84(c),
                                                                                         FUN_0001ba18:0001bb00(c),
                                                                                         FUN_0001ba18:0001bb7c(c),
                                                                                         FUN_0001ba18:0001bf80(c),
                                                                                         FUN_0001ba18:0001bc54(c),
                                                                                         FUN_0001ba18:0001bcd0(c),
                                                                                         FUN_0001ba18:0001bd6c(c),
                                                                                         FUN_0001ba18:0001be80(c),
                                                                                         FUN_0003319c:000331e4(c), [more]

000410c0 30 00 00 f0     adrp        x16,0xA0000
000410c4 11 4a 4a f9     ldr         x17,[x16, #0x890]=>PTR_00048090                    = 00041070
000410c8 10 42 22 91     add         x16,x16,#0x890
000410cc 20 02 1f d6     br          x17=>SUB_ffffffffffffa1070
                         -- Flow Override: CALL_RETURN (COMPUTED_CALL_TERMINATOR)

                         **************************************************************
                         *                          THUNK FUNCTION                    *
                         **************************************************************
                         thunk undefined thunk_EXT_FUN_00041070()
                           Thunked-Function: <EXTERNAL>::EXT_FUN_000...
     undefined           w0:1                <RETURN>
                         <EXTERNAL>::thunk_EXT_FUN_00041070                 XREF[37]:    FUN_00017a8c:00017ac8(c),
                                                                                         FUN_00017ad8:00017dc4(c),
                                                                                         FUN_00019f38:0001a0a0(c),
                                                                                         FUN_00019f38:0001a1b0(c),
                                                                                         FUN_00019f38:0001a3a8(c),
                                                                                         FUN_00019f38:0001a530(c),
                                                                                         FUN_00019f38:0001a6ac(c),
                                                                                         FUN_00019f38:0001a888(c),
                                                                                         FUN_0001ba18:0001bd78(c),
                                                                                         FUN_0001ba18:0001bf60(c),
                                                                                         FUN_0001ba18:0001c1a0(c),
                                                                                         FUN_0001ba18:0001c42c(c),
                                                                                         FUN_00022208:000224f0(c),
                                                                                         FUN_00024114:00024414(c),
                                                                                         FUN_00024114:000247c0(c),
```



Conclusion

Conclusion

- Executable file formats modifications (still) have an impact on all the reverse engineering tools.
- This is a topic that is less explored than regular obfuscation.

Conclusion

- Executable file formats modifications (still) have an impact on all the reverse engineering tools.
- This is a topic that is less explored than regular obfuscation.
- $\Rightarrow$ less covered by recovering *scripts* and papers.

Conclusion

- Executable file formats modifications (still) have an impact on all the reverse engineering tools.
- This is a topic that is less explored than regular obfuscation.
- $\Rightarrow$ less covered by recovering *scripts* and papers.
- Can be used in pair with *classical* obfuscation.

Thank you for your attention

https://github.com/romainthomas/the-poor-mans-obfuscator

https://www.romainthomas.fr/publication

Thank you for your attention

https://github.com/romainthomas/the-poor-mans-obfuscator

https://www.romainthomas.fr/publication

Thank you for your attention

https://github.com/romainthomas/the-poor-mans-obfuscator

https://www.romainthomas.fr/publication

Questions?