exness

mwemu

radare2 capabilities inside mwemu emulated process.

r2con2025

@sha0coder

October 2025

Index

1. mwemu project

- project
- ecosystem
- collaborators

2. Windows process internals

- how a process looks like from inside
- address space, modules and structures

3. Example with enigma packer

- emulating some millions of instructions
- capturing a specific instant

4. Radare2 inside

- viewing data blobs
- code static-analysis
- radare2 integration
- future improvements

5. Demo

- mwemu, radare2 and enigma packer in action.

exness

1. mwemu project

mwemu

- started on October 2021
- main purpose: controlling algorithms & predicting things.
- HW emulation (x86) + OS simulation (windows mainly)
- 32+64bits implemented from scratch (exced bytes to asm done with iced-x86)
- not based on unicorn, qemu or any emulation lib, 100% pure rust.
- open source: https://github.com/sha0coder/mwemu
- renamed from scemu to mwemu.

mwemu

HW Emulation

- x86 32 and 64bits on same engine.
- registers, flags, FPU, f80, interrupts.
- 364 instructions implemented.
- logic operations, rotation, shifts, etc.
- exotic instructions like xmm and ymm
- jump, call, branches
- gateway to the OS

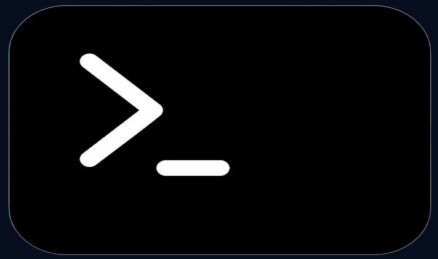
OS Simulation

- file loaders PE, PE+, ELF64, shellcodes
- iat binding, crafting peb,teb,ldr
- allocators
- linux64 syscalls (basic support and only for static)
- exception (SEH, VEH and UEF)
- 615 winapi implemented
- generic emulation for unimplemented winapis

mwemu ecosystem

- mwemu - command line
- pymwemu - python3
- libmwemu - rust lib on crates.io

(linux & windows & mac)



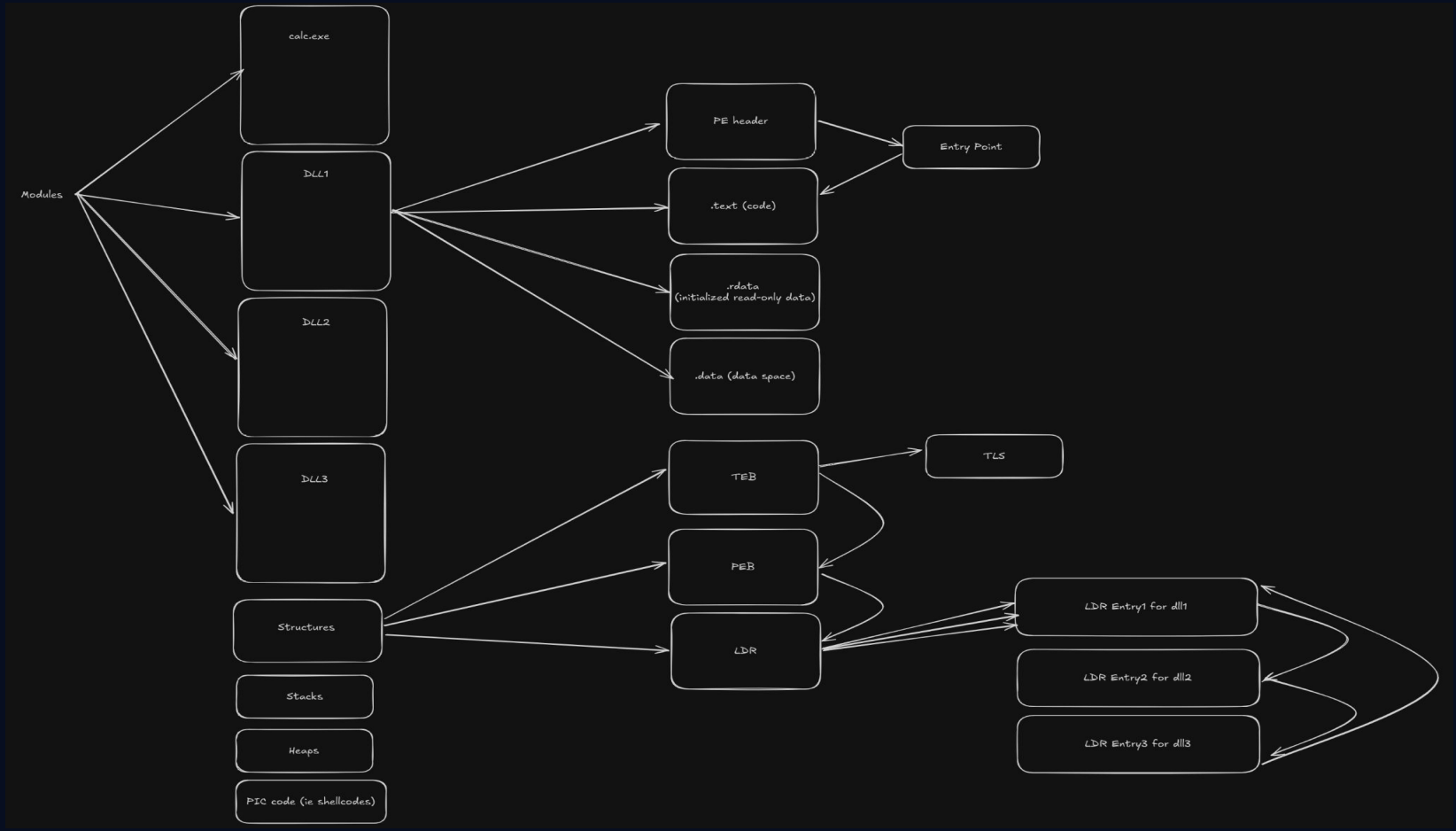
mwemu - contributors

- brandonros based on America
- acheron2302 based on Asia
- elcapor based on unknown
- sha0coder (creator) based on Europe

exness

2. Windows Process internals

Windows Process Internals



exness

mwemu virtual memory

```
--- console ---
=>m
print_maps
--- maps ---
ntdll.mrdata        0x70124000 - 0x70127000 (12288)
peb                 0x70001000 - 0x70001320 (800)
ntdll.rsrc          0x70128000 - 0x70198000 (458752)
RtlUserProcessParameters32 0x60001000 - 0x60001048 (72)
ldr                 0x70000000 - 0x70000094 (148)
kernelbase.text     0x7027e000 - 0x70454000 (1925120)
stack               0x212000 - 0x242000 (196608)
teb                 0x70002000 - 0x700023e8 (1000)
command_line        0x60003000 - 0x6000306e (110)
ntdllRT             0x7011d000 - 0x7011e000 (4096)
kernelbase.rsrc     0x7045f000 - 0x70460000 (4096)
kernel32.data       0x7024d000 - 0x7025d000 (65536)
file_name           0x60002000 - 0x6000206c (108)
kernelbase.didat    0x7045e000 - 0x7045f000 (4096)
kernel32.reloc      0x7026d000 - 0x7027d000 (65536)
ntdll.pe            0x70003000 - 0x70004000 (4096)
ntdll.text          0x70004000 - 0x7011d000 (1150976)
ntdll.00cfg         0x70127000 - 0x70128000 (4096)
ntdll.dll.ldr       0x60004000 - 0x60004888 (2184)
kernelbase.reloc    0x70460000 - 0x70490000 (196608)
kernelbase.idata    0x70458000 - 0x7045e000 (24576)
loader.exe.ldr      0x60000000 - 0x60000888 (2184)
kernel32.rdata      0x7021d000 - 0x7024d000 (196608)
kernel32.rsrc       0x7025d000 - 0x7026d000 (65536)
kernel32.dll.ldr    0x60005000 - 0x60005888 (2184)
kernel32.text       0x701ad000 - 0x7021d000 (458752)
kernel32.pe         0x7019d000 - 0x701ad000 (65536)
ntdll.data          0x7011e000 - 0x70124000 (24576)
kernelbase.dll.ldr  0x60006000 - 0x60006888 (2184)
kernelbase.data     0x70454000 - 0x70458000 (16384)
memory usage: 21984918 bytes
=>mt
mem test passed ok.
```

```
=>ldr
0x7fe000000000 loader.exe flink:7fe000004000 blink:7fe0003ec000 base:140000000 pe_hdr:e8 5045
0x7fe000004000 ntdll.dll flink:7fe000005000 blink:7fe000000000 base:7ff000003000 pe_hdr:e8 5045
0x7fe000005000 kernel32.dll flink:7fe000006000 blink:7fe000004000 base:7ff0001f8000 pe_hdr:f0 5045
0x7fe000006000 kernelbase.dll flink:7fe000007000 blink:7fe000005000 base:7ff0002b5000 pe_hdr:f0 5045
0x7fe000007000 iphlpapi.dll flink:7fe000008000 blink:7fe000006000 base:7ff00057e000 pe_hdr:f8 5045
0x7fe000008000 ws2_32.dll flink:7fe000009000 blink:7fe000007000 base:7ff0005b8000 pe_hdr:f0 5045
0x7fe000009000 advapi32.dll flink:7fe00000a000 blink:7fe000008000 base:7ff000623000 pe_hdr:100 5045
0x7fe00000a000 comctl32.dll flink:7fe00000b000 blink:7fe000009000 base:7ff0006cd000 pe_hdr:f0 5045
0x7fe00000b000 winhttp.dll flink:7fe00000c000 blink:7fe00000a000 base:7ff00095f000 pe_hdr:f8 5045
0x7fe00000c000 wininet.dll flink:7fe00000d000 blink:7fe00000b000 base:7ff000a4f000 pe_hdr:f0 5045
0x7fe00000d000 dnsapi.dll flink:7fe00000e000 blink:7fe00000c000 base:7ff000cd9000 pe_hdr:f8 5045
0x7fe00000e000 shell32.dll flink:7fe00000f000 blink:7fe00000d000 base:7ff000da4000 pe_hdr:f0 5045
0x7fe00000f000 shlwapi.dll flink:7fe0003e1000 blink:7fe00000e000 base:7ff001b2c000 pe_hdr:f0 5045
0x7fe0003e1000 user32.dll flink:7fe0003e2000 blink:7fe00000f000 base:7ff001b7e000 pe_hdr:f0 5045
0x7fe0003e2000 oleaut32.dll flink:7fe0003e3000 blink:7fe0003e1000 base:7ff001d13000 pe_hdr:f8 5045
0x7fe0003e3000 gdi32.dll flink:7fe0003e4000 blink:7fe0003e2000 base:7ff001dd8000 pe_hdr:f0 5045
0x7fe0003e4000 version.dll flink:7fe0003e5000 blink:7fe0003e3000 base:7ff001e02000 pe_hdr:e0 5045
0x7fe0003e5000 ole32.dll flink:7fe0003e6000 blink:7fe0003e4000 base:7ff001e0e000 pe_hdr:f8 5045
0x7fe0003e6000 api-ms-win-core-synch-l1-2-0.dll flink:7fe0003e7000 blink:7fe0003e5000 base:7ff001f65000 pe_hdr:d0 5045
0x7fe0003e7000 vcruntime140.dll flink:7fe0003e8000 blink:7fe0003e6000 base:7ff001f68000 pe_hdr:f0 5045
0x7fe0003e8000 api-ms-win-crt-runtime-l1-1-0.dll flink:7fe0003e9000 blink:7fe0003e7000 base:7ff001f81000 pe_hdr:d0 5045
0x7fe0003e9000 api-ms-win-crt-math-l1-1-0.dll flink:7fe0003ea000 blink:7fe0003e8000 base:7ff001f85000 pe_hdr:d0 5045
0x7fe0003ea000 api-ms-win-crt-stdio-l1-1-0.dll flink:7fe0003eb000 blink:7fe0003e9000 base:7ff001f8a000 pe_hdr:d0 5045
0x7fe0003eb000 api-ms-win-crt-locale-l1-1-0.dll flink:7fe0003ec000 blink:7fe0003ea000 base:7ff001f8e000 pe_hdr:d0 5045
0x7fe0003ec000 api-ms-win-crt-heap-l1-1-0.dll flink:7fe000000000 blink:7fe0003eb000 base:7ff001f91000 pe_hdr:d0 5045
=>
```

exness

3. Example with Enigma Packer

Enigma Packer - emulating first loop

```
~/s/mwemu ❯❯❯ cargo run --release -- -f ~/Downloads/Telegram\ Desktop/enigma_test_protected.exe -6 -c 100000 -vv
```

- First millions of instruction is a XOR loop.
- Loop ends in moment: 230,523,805
- why? code protection or anti-emulation?
- it takes just 11.71 seconds for mwemu.

```
99901 0x14104a3c0: jne   0000000014104A3B4h taken
99902 0x14104a3b4: xor   [rcx],edx
99903 0x14104a3b6: add   rcx,4
99904 0x14104a3bd: dec   rax
99905 0x14104a3c0: jne   0000000014104A3B4h taken
99906 0x14104a3b4: xor   [rcx],edx
99907 0x14104a3b6: add   rcx,4
99908 0x14104a3bd: dec   rax
99909 0x14104a3c0: jne   0000000014104A3B4h taken
99910 0x14104a3b4: xor   [rcx],edx
99911 0x14104a3b6: add   rcx,4
99912 0x14104a3bd: dec   rax
99913 0x14104a3c0: jne   0000000014104A3B4h taken
99914 0x14104a3b4: xor   [rcx],edx
99915 0x14104a3b6: add   rcx,4
99916 0x14104a3bd: dec   rax
99917 0x14104a3c0: jne   0000000014104A3B4h taken
99918 0x14104a3b4: xor   [rcx],edx
99919 0x14104a3b6: add   rcx,4
99920 0x14104a3bd: dec   rax
99921 0x14104a3c0: jne   0000000014104A3B4h taken
99922 0x14104a3b4: xor   [rcx],edx
99923 0x14104a3b6: add   rcx,4
99924 0x14104a3bd: dec   rax
99925 0x14104a3c0: jne   0000000014104A3B4h taken
99926 0x14104a3b4: xor   [rcx],edx
99927 0x14104a3b6: add   rcx,4
99928 0x14104a3bd: dec   rax
99929 0x14104a3c0: jne   0000000014104A3B4h taken
99930 0x14104a3b4: xor   [rcx],edx
99931 0x14104a3b6: add   rcx,4
99932 0x14104a3bd: dec   rax
99933 0x14104a3c0: jne   0000000014104A3B4h taken
99934 0x14104a3b4: xor   [rcx],edx
99935 0x14104a3b6: add   rcx,4
99936 0x14104a3bd: dec   rax
99937 0x14104a3c0: jne   0000000014104A3B4h taken
99938 0x14104a3b4: xor   [rcx],edx
99939 0x14104a3b6: add   rcx,4
99940 0x14104a3bd: dec   rax
99941 0x14104a3c0: jne   0000000014104A3B4h taken
99942 0x14104a3b4: xor   [rcx],edx
99943 0x14104a3b6: add   rcx,4
99944 0x14104a3bd: dec   rax
99945 0x14104a3c0: jne   0000000014104A3B4h taken
99946 0x14104a3b4: xor   [rcx],edx
99947 0x14104a3b6: add   rcx,4
99948 0x14104a3bd: dec   rax
99949 0x14104a3c0: jne   0000000014104A3B4h taken
99950 0x14104a3b4: xor   [rcx],edx
99951 0x14104a3b6: add   rcx,4
99952 0x14104a3bd: dec   rax
99953 0x14104a3c0: jne   0000000014104A3B4h taken
99954 0x14104a3b4: xor   [rcx],edx
99955 0x14104a3b6: add   rcx,4
99956 0x14104a3bd: dec   rax
99957 0x14104a3c0: jne   0000000014104A3B4h taken
99958 0x14104a3b4: xor   [rcx],edx
99959 0x14104a3b6: add   rcx,4
99960 0x14104a3bd: dec   rax
99961 0x14104a3c0: jne   0000000014104A3B4h taken
99962 0x14104a3b4: xor   [rcx],edx
```

exness

Enigma Packer – first api call

```
~/s/mwemu ❯❯❯ time cargo run --release -- -f ~/Downloads/Telegram\ Desktop/enigma_test_protected.exe -6 -c 230523814 --cmd 'q'
```

```
230523805 0x14104a774: jne   000000014104A76Ch not taken
230523806 0x14104a77a: jmp   000000014104A783h
       mem_trace: pos = 230523807 rip = 14104a783 op = read bits = 32 address = 0x14002a03c value = 0x80 name = 'enigma_test_protected2a000'
230523807 0x14104a783: mov   edi,[rsi+3Ch] ; 0x80
       mem_trace: pos = 230523808 rip = 14104a786 op = read bits = 32 address = 0x14002a110 value = 0xcd8000 name = 'enigma_test_protected2a000'
230523808 0x14104a786: mov   edi,[rdi+rsi+90h] ; 0xcd8000
230523809 0x14104a78d: add   rdi,rsi
230523810 0x14104a790: cmp   dword ptr [rdi+0Ch],0
       mem_trace: pos = 230523810 rip = 14104a790 op = read bits = 32 address = 0x140d0200c value = 0xcd92cc name = 'enigma_test_protected2a000'
       cmp: 0xcd92cc > 0x0
230523811 0x14104a794: je    000000014104A838h not taken
       mem_trace: pos = 230523812 rip = 14104a79a op = read bits = 32 address = 0x140d0200c value = 0xcd92cc name = 'enigma_test_protected2a000'
230523812 0x14104a79a: mov   ecx,[rdi+0Ch] ; 0xcd92cc
230523813 0x14104a79d: add   rcx,rsi
-------
230523814 0x14104a7a0: call  qword ptr [rbp+0D2511Ch]
--- console ---
=>
230523814 0x14104a7a0: call  qword ptr [rbp+0D2511Ch]
       mem_trace: pos = 230523814 rip = 14104a7a0 op = read bits = 64 address = 0x140d2511c value = 0x7ff000217050 name = 'enigma_test_protectedd21000'
       mem_trace: pos = 230523814 rip = 14104a7a0 op = write bits = 64 address = 0x329f58 value = 0x14104a7a6 name = 'stack'
/!\ changing RIP to kernel32.text
       mem_trace: pos = 230523814 rip = 14104a7a0 op = read bits = 64 address = 0x329f50 value = 0x14104a7a6 name = 'stack'
       mem_trace: pos = 230523814 rip = 14104a7a0 op = write bits = 64 address = 0x329f50 value = 0x14104a7a6 name = 'register'
** 230523814:14104a7a6 kernel32!GetModuleHandleA `kernel32.dll` 7ff0001f8000
=>q
```

exness

Trying full-emulation first

```
~/s/mwemu ❯❯❯ cargo run --release -- -f ~/Downloads/Telegram\ Desktop/enigma_test_protected.exe -6
```

```
** 238232127:140c1eb42 kernel32!GetCommandLineA
** 238234333:14004023c kernel32!GetLastError:0
** 238234339:14004024c ** 238234339 kernel32!TlsGetValue idx: 1 =0x7fe000012000
** 238234346:14004027b kernel32!SetLastError err: 0
** 238236568:140c1eb60 kernel32!GetCommandLineW
** 238238157:140c1eb74 kernel32!GetCommandLineW
** 238239658:1400368d2 :1400368d2 oleaut32!SysAllocStringLen str_ptr: 0x0 size: 10
** 238239658:1400368d2 SysAllocStringLen returning: 0x7fe001423004 (base: 0x7fe001423000, length_prefix: 20)
** 238242485:140c1eb88 kernel32!GetModuleHandleA `ntdll.dll` 7ff000003000
** 238247870:1400368d2 :1400368d2 oleaut32!SysAllocStringLen str_ptr: 0x0 size: 27
** 238247870:1400368d2 SysAllocStringLen returning: 0x7fe001424004 (base: 0x7fe001424000, length_prefix: 54)
** 238251222:1400368d2 :1400368d2 oleaut32!SysAllocStringLen str_ptr: 0x0 size: 6
** 238251222:1400368d2 SysAllocStringLen returning: 0x7fe001425004 (base: 0x7fe001425000, length_prefix: 12)
** 238255518:140c16c6c ntdll!RtlDosPathNameToNtPathName_U dos_path='c:\cwd' dos_path_name_ptr: 0x7fe001425004 nt_path_name_ptr: 0x329730 nt_file_name_
ir_ptr: 0x0
** 238255518:140c16c6c Converted DOS path 'c:\cwd' to NT path '\??\c:\cwd'
** 238255518:140c16c6c Created UNICODE_STRING: Length=20, MaxLength=22, Buffer=0x7fe001426000
** 238264177:1400c46c0 ** 238264177 ntdll!NtCreateFile | Handle=0x329750 Access=0x80100080 ObjAttr=0x329700 IoStat=0x329740 AllocSz=0x0 FileAttr=0x80
eateDisp=0x1 CreateOpt=0x60 EaBuff=0x0 EaLen=0x0
** 238264177:1400c46c0 ** 238264177 Reading OBJECT_ATTRIBUTES at 0x329700
** 238264177:1400c46c0 ** 238264177 OBJECT_ATTRIBUTES structure dump:
** 238264177:1400c46c0 ** 238264177   +0x00: 0x00007ff000000030
** 238264177:1400c46c0 ** 238264177   +0x08: 0x0000000000000000
** 238264177:1400c46c0 ** 238264177   +0x10: 0x0000000000329730
** 238264177:1400c46c0 ** 238264177   +0x18: 0x0000000000000040
** 238264177:1400c46c0 ** 238264177   +0x20: 0x0000000000000000
** 238264177:1400c46c0 ** 238264177   +0x28: 0x0000000000000000
** 238264177:1400c46c0 ** 238264177 RootDirectory: 0x0, ObjectName pointer: 0x329730
** 238264177:1400c46c0 ** 238264177 Reading UNICODE_STRING at 0x329730
** 238264177:1400c46c0 ** 238264177 UNICODE_STRING +0x00: 0x0000000100160014
** 238264177:1400c46c0 ** 238264177 UNICODE_STRING +0x08: 0x00007fe001426000
** 238264177:1400c46c0 ** 238264177 UNICODE_STRING: Length=20 MaxLength=22 Buffer=0x7fe001426000
** 238264177:1400c46c0 ** 238264177 Filename: '\??\c:\cwd'
** 238264177:1400c46c0 ** 238264177 ntdll!NtCreateFile resolved filename: '\??\c:\cwd'
** 238266056:140c16c8a ntdll!RtlFreeAnsiString 0x3315504
** 238268776:140036959 oleaut32!SysFreeString  0x7fe001425004
** 238268776:140036959 SysFreeString zeroing 18 bytes starting at 0x7fe001425000 (string data was 12 bytes, 6 chars)
** 238269622:140036959 oleaut32!SysFreeString  0x7fe001424004
** 238269622:140036959 SysFreeString zeroing 60 bytes starting at 0x7fe001424000 (string data was 54 bytes, 27 chars)
** 238282009:1400c44a4 ntdll!NtSetInformationFile handle: 0x34 info_class: 14 length: 8
** 238297408:1400c45ec ntdll!NtReadFile \??\c:\cwd hndl: 0x34 buff: 0x329c10 sz: 64 off_var: 0x0
=== EMULATOR STATE AT PANIC ===
Current position: 238297408
238297408 0x1400c45e6: call  qword ptr [140269710h]
Registers:
  RAX: 0x0000000000329730  RBX: 0x00007fe000f0ed50
  RCX: 0x0000000000000034  RDX: 0x0000000000000000
  RSI: 0x000000000000000a  RDI: 0x0000000140d020f0
  RBP: 0x00000000003296c8  RSP: 0x0000000000329648
  R8:  0x0000000000000000  R9:  0x0000000000000000
  R10: 0x00650076005c0000  R11: 0x0000000140290578
  R12: 0x0000000140d032c4  R13: 0x00007ff001f94000
  R14: 0x0000000000000000  R15: 0x0000000000000000
  RIP: 0x0000001400c45ec
EFLAGS: 0x00000046
Last instruction:
Instruction size: 0
Call stack (last 1 entries):
  0: 1400c45e6:call:7ff00009ee70
Tick count: 94
Base address: 0x140000000
Filename: /home/uid0/Downloads/Telegram Desktop/enigma_test_protected.exe
============================
```

- The problem is not x86 emulation.
- It's windows simulation.
- full-emulation is only for simpler packers and shellcode encoders

- pymwemu Resulted very useful
- python module for emulating

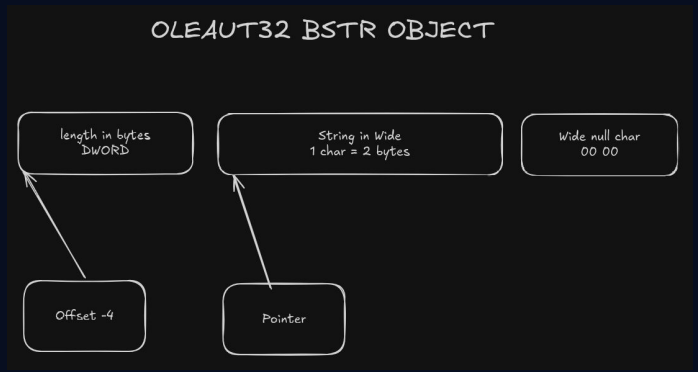
exness

exness

4. radare2 inside

radare2 inside – viewing data blobs

```
-------
238217842 0x140d06de1: jmp   qword ptr [140D026C4h]
--- console ---
=>r rcx
       rcx: 0x3297b8 3315640 (stack)
=>r rdx
       rdx: 0x7fe00141c004 140600070488068 'c:\cwd' (bstr_7fe00141c000)
=>r r8
       r8 : 0x26 38
=>
238217842 0x140d06de1: jmp   qword ptr [140D026C4h]
       mem_trace: pos = 238217842 rip = 140d06de1 op = read bits = 64 address = 0x140d026c4 value = 0x7ff001d20f40 name = 'enigma_test_protected2a000'
/!\ changing RIP to oleaut32.text
       mem_trace: pos = 238217842 rip = 140d06de1 op = read bits = 64 address = 0x3296c0 value = 0x140036d0c name = 'stack'
       mem_trace: pos = 238217842 rip = 140d06de1 op = write bits = 64 address = 0x3296c0 value = 0x140036d0c name = 'register'
** 238217842:140036d0c oleaut32!SysReAllocStringLen pbstr_ptr: 0x3297b8 psz: 0x7fe00141c004 len: 38
** 238217842:140036d0c Old BSTR content: "C:\cwd" (length: 38 chars)
** 238217842:140036d0c New source string: "c:\cwd" (length: 38 chars)
** 238217842:140036d0c Final BSTR content: "c:\cwd" (length: 38 chars)
** 238217842:140036d0c oleaut32!SysReAllocStringLen allocated new string at 0x7fe00141d004 size: 76 (base: 0x7fe00141d000)
=>
```



exness

radare2 inside – viewing data blobs

```
** 238217842:140036d0c oleaut32!SysReAllocStringLen pbstr_ptr: 0x3297b8 psz: 0x7fe00141c004 len: 38
** 238217842:140036d0c oleaut32!SysReAllocStringLen Old BSTR content: "C:\cwd" (length: 38 chars)
** 238217842:140036d0c oleaut32!SysReAllocStringLen New source string: "c:\cwd" (length: 38 chars)
** 238217842:140036d0c oleaut32!SysReAllocStringLen Final BSTR content: "c:\cwd" (length: 38 chars)
** 238217842:140036d0c oleaut32!SysReAllocStringLen allocated new string at 0x7fe00141d004 size: 76 (base: 0x7fe00141d000)
=>r2 0x7fe00141d004
spawning radare2 software.
 -- This binary may contain traces of human
[0x7fe00141d004]> px
- offset -       4 5  6 7  8 9  A B  C D  E F 1011 1213  456789ABCDEF0123
0x7fe00141d004  6300 3a00 5c00 6300 7700 6400 0000 5c00  c.:.\.c.w.d...\.
0x7fe00141d014  6300 3a00 5c00 7500 7300 6500 7200 7300  c.:.\.u.s.e.r.s.
0x7fe00141d024  5c00 7500 7300 6500 7200 5c00 6100 7000  \.u.s.e.r.\.a.p.
0x7fe00141d034  7000 6400 6100 7400 6100 5c00 7200 6f00  p.d.a.t.a.\.r.o.
0x7fe00141d044  6100 6d00 6900 6e00 6700 5c00 0000 0000  a.m.i.n.g.\.....
0x7fe00141d054  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x7fe00141d064  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x7fe00141d074  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x7fe00141d084  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x7fe00141d094  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x7fe00141d0a4  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x7fe00141d0b4  0000 ffff ffff ffff ffff ffff ffff ffff  ................
0x7fe00141d0c4  ffff ffff ffff ffff ffff ffff ffff ffff  ................
0x7fe00141d0d4  ffff ffff ffff ffff ffff ffff ffff ffff  ................
0x7fe00141d0e4  ffff ffff ffff ffff ffff ffff ffff ffff  ................
0x7fe00141d0f4  ffff ffff ffff ffff ffff ffff ffff ffff  ................
[0x7fe00141d004]> px -4
- offset -       0 1  2 3  4 5  6 7  8 9  A B  C D  E F  0123456789ABCDEF
0x7fe00141d000  4c00 0000                               L...
[0x7fe00141d004]> pv4 -4
0x0000004c
[0x7fe00141d004]> ? 0x4c
int32   76
uint32  76
hex     0x4c
octal   0114
unit    76
segment 0000:004c
string  "L"
fvalue  76.0
float   0.000000000000000f
double  0.000000000000000
binary  0b01001100
base36  0_24
ternary 0t2211
[0x7fe00141d004]>
```

exness

radare2 inside – static analysis and AI

```
[0x140d06de1 | fcn.140d06de1]> afl | wc -l
13437
[0x140d06de1 | fcn.140d06de1]> pdf
           ;-- rip:
┌ 6: fcn.140d06de1 ();
└           0x140d06de1      ff25ddb8ffff   jmp qword [0x140d026c4]    ; [0x140d026c4:8]=0x7ff001d20f40
[0x140d06de1 | fcn.140d06de1]> pdc
// callconv: rax amd64 (rdi, rsi, rdx, rcx, r8, r9, xmm0, xmm1, xmm2, xmm3, xmm4);
int fcn.140d06de1 (int rdi, int rsi) {
    loc_0x140d06de1:
        goto loc_qword [0x140d026c4] // [0x140d026c4:8]=0x7ff001d20f40
        return rax;
}

[0x140d06de1 | fcn.140d06de1]> axt @fcn.140d06de1
[0x140d06de1 | fcn.140d06de1]> axt @0x140d06de1
[0x140d06de1 | fcn.140d06de1]> axt 0x140d06de1
[0x140d06de1 | fcn.140d06de1]> s 0x14002BB40h
[0x14002bb40 | fcn.14002bb40]> decai -d
The provided output from `pdd` and `pdg` shows a function `fcn_14002bb40` in what appears to be a decompiled or p
 (Radare2)**. The function is very minimal and contains only a call to some address (`0x1403b485c`) which is trea

Here's how you can translate this into **Python**:

### ✅ Python Translation

def fcn_14002bb40():
    # WARNING: Could not recover jumptable at 0x00014002bb40. Too many branches
    # WARNING: Treating indirect jump as call
    # Indirect call to address 0x1403b485c
    # In Python, we simulate this by calling a function at that address (if known)
    # Since we don't have the actual function, we just return
    return

### 🔍 Explanation:
- The original C-style pseudo-code shows a function that does nothing but calls another function indirectly via a
- In Python, since we don't have access to raw memory addresses or actual machine code execution, we simulate the
., just returns).
- If you were to actually execute this code in a real environment (like a debugger or emulator), you'd need to re
can't do that without more context or tools like `capstone`, `unicorn`, etc.

---

### 🐑 Optional Enhancement (if you have a known target function):

If you know what function `0x1403b485c` points to, you could replace the `return` with a call to that function:

def some_target_function():
    # Placeholder for actual logic
    pass

def fcn_14002bb40():
    some_target_function()
    return

Let me know if you want help translating more complex logic or integrating with disassembly tools!
[0x14002bb40 | fcn.14002bb40]>
```

exness

radare2 inside – static analysis and AI

```
[0x140040220 | fcn.140040220]> decai -d
Here's the **Python translation** of the provided pseudo-code from the `r2dec` output. The function `fcn_140040220` is translated into Python, preserving its logi
nd structure.

### ✔  Python Equivalent:

def fcn_140040220(arg1, arg2, arg4):
    # Simulate stack frame setup (not needed in Python but for clarity)
    # In real assembly, this would involve stack manipulation

    # Call fcn_14002b030()
    esi = fcn_14002b030()

    # Dereference pointer at 0x140258480 and call fcn_14002b020 on it
    rax = fcn_14002b020(**0x140258480)  # This assumes **0x140258480 is a valid pointer to a function

    if rax == 0:
        fcn_140040100()
        rax = fcn_14002b020(**0x140258480)
        fcn_14003fab0(0x1000000)

    # Call fcn_14002b040 with esi as argument
    fcn_14002b040(esi)

    # Return value: (arg1 & 0xFFFFFFFF) + rax
    return (arg1 & 0xFFFFFFFF) + rax

---

### ☯ Notes:

- `fcn_14002b030`, `fcn_14002b020`, `fcn_140040100`, `fcn_14003fab0`, and `fcn_14002b040` are assumed to be defined elsewhere in your codebase or libraries.
- The `**0x140258480` syntax implies that `0x140258480` is a pointer to a pointer (or a function pointer), which is typical in low-level code.
- The `& 0xFFFFFFFF` operation ensures we're working with a 32-bit value (as in x86-64 where `arg1` is 64-bit but we mask it to 32-bit).
- This function returns a sum of a masked version of `arg1` and the result of `fcn_14002b020`.

Let me know if you'd like a version that simulates the actual memory or function calls using `ctypes` or similar!
[0x140040220 | fcn.140040220]>
```

exness

radare2 inside – integration

```rust
pub fn spawn_radare2(addr: u64, emu: &mut Emu) {
    let mem = match emu.maps.get_mem_by_addr(addr) {
        Some(m) => m,
        None => {
            log::info!("address not found on any map");
            return;
        }
    };

    let tmpfile = format!("/tmp/{}.r2", mem.get_name());
    mem.save_all(&tmpfile);

    let base = format!("0x{:x}", mem.get_base());
    let seek = format!("0x{:x}", addr);
    let bits;
    if emu.cfg.is_64bits {
        bits = "64"
    } else {
        bits = "32"
    }
    let precmd = format!("dr rax={}?; dr rbx={}?; dr rcx={}?; dr rdx={}?; dr rsi={}?;
                          dr rdi={}?; dr rbp={}?; dr rsp={}?; dr rip={}?; dr r8={}?
                          dr r9={}?; dr r10={}?; dr r11={}?; dr r12={}?; dr r13={}?;
                          dr r14={}?; dr r15={}?; decai -e model=qwen3-coder:30b; r2ai -e r2ai.model=qwen3-coder:30b;",
                          emu.regs().rax, emu.regs().rbx, emu.regs().rcx, emu.regs().rdx,
                          emu.regs().rsi, emu.regs().rdi, emu.regs().rbp, emu.regs().rsp,
                          emu.regs().rip, emu.regs().r8, emu.regs().r9, emu.regs().r10,
                          emu.regs().r11, emu.regs().r12, emu.regs().r13, emu.regs().r14,
                          emu.regs().r15);
    let r2args = vec![
        "-n", "-a", "x86", "-b", &bits, "-m", &base, "-s", &seek, "-c", &precmd, &tmpfile,
    ];

    log::info!("spawning radare2 software.");

    match Command::new("radare2")
        .args(&r2args)
        .stdin(Stdio::inherit())
        .stdout(Stdio::inherit())
        .stderr(Stdio::inherit())
        .spawn()
    {
        Ok(mut child) => {
            let _ = child.wait();
        }
        Err(e) => {
            log::error!("Install radare first! {}", e);
            return;
        }
    }

    if let Err(e) = fs::remove_file(&tmpfile) {
        if e.kind() != io::ErrorKind::NotFound {
            log::error!("temporal file not found");
        }
    }
}
```

exness

exness

5. demo

Thank you!

https://github.com/sha0coder/mwemu
Jesus Olmos
@sha0coder