

Programando Assembly no Linux x64

+ O arquivo com nossas syscalls será o

/usr/include/x86_64-linux-gnu/asm/unistd_64.h

→ dessa vez a write será setada como 1

→ invés da interrupção 0x80, usaremos **syscall**

→ site para arquitetura x64: https://syscalls.33challs.com/?arch=x86_64

-----ass2.asm-----

global _main

section .data

curso: db "DeSec Security", 0xa

section .text

_main:

mov rax, 1 ; a write em 64 bits é 1

mov rdi, 1 ; nosso file descriptor que é 1 tbm

mov rsi, curso

mov rdx, 15 ; tamanho

syscall

mov rax, 60; a exit aq é 60

mov edi, 0

syscall

```
nasm -f elf64 ass2.asm
```

```
file ass2.o
```

ass2.o: ELF 64-bit LSB relocatable, x86-64, version 1 (SYSV), not stripped

→ Vemos que de fato ele gerou um 64-bit

```
objdump -d -M intel ass2.o
```

```
ass2.o: file format elf64-x86-64
```

```
Disassembly of section .text:
```

```
0000000000000000 <_main>:
```

```
0: b8 01 00 00 00 00 74 [2] mov    eax,0x1
5: bf 01 00 00 00 00 77 [2] mov    edi,0x1
a: 48 be 00 00 00 00 00 [2] movabs rsi,0x0
11: 00 00 00 00 00 00 78 [2] 
14: ba 0f 00 00 00 00 71 [2] mov    edx,0xf
19: 0f 05 [2] syscall
1b: b8 3c 00 00 00 00 74 [2] mov    eax,0x3c
20: bf 00 00 00 00 00 03 [2] mov    edi,0x0
25: 0f 05 [2] syscall
```

→ Por mais que tenhamos usado um formato 64-bit, ele otimizou para apenas 32-bit pelo motivo de termos feito uso de pouca informação

→ Se quisermos que ele gere de fato um 64-bits:

```
(root@DESKTOP-NJHHNK6)-[/home/kali] Node.js
# nasm -O0 -f elf64 ass2.asm [debug] autosave
[2024-05-01 21:28:08.527] [ ] [debug] autosave
(root@DESKTOP-NJHHNK6)-[/home/kali] autosave
# objdump -d -M intel ass2.o [debug] autosave
[2024-05-01 21:31:08.522] [ ] [debug] autosave
ass2.o: file format elf64-x86-64
[2024-05-01 21:33:08.471] [ ] [debug] autosave
[2024-05-01 21:34:08.471] [ ] [debug] autosave
Disassembly of section .text:
[2024-05-01 21:36:08.474] [ ] [debug] autosave
0000000000000000 <_main>:
[200:-05 48 b8 01 00 00 00 00] movabs rax,0x1
[207:-05 00 00 00 00 08.486] [ ] [debug] autosave
[20a:-05 48 bf 01 00 00 00 00] movabs rdi,0x1
[211:-05 00 00 00 00 08.506] [ ] [debug] autosave
[214:-05 48 be 00 00 00 00 00] movabs rsi,0x0
[21b:-05 00 00 00 00 08.530] [ ] [debug] autosave
[21e:-05 48 ba 0f 00 00 00 00] movabs rdx,0xf
[225:-05 00 00 00 00 08.522] [ ] [debug] autosave
[228:-05 0f 05 00 00 00 00] syscall
[22a:-05 48 b8 3c 00 00 00 00] movabs rax,0x3c
[231:-05 00 00 00 00 08.471] [ ] [debug] autosave
[234:-05 bf 00 00 00 00 00 1] mov edi,0x0
[239:-05 0f 05 00 00 00 00] syscall
[240:-05 00 00 00 00 08.471] [ ] [debug] autosave
```

```
ld --entry main ass2.o -o ass2
```

```
[2024-05-01 21:45:08.522] [ ] [debug] au
└─(root@DESKTOP-NJHHNK6)-[/home/kali]
└─# ld --entry: _main ass2.o -o ass2
[2024-05-01 21:48:08.471] [ ] [debug] au
└─(root@DESKTOP-NJHHNK6)-[/home/kali]
└─# ./ass2
Desec Security: 51:08.471] [ ] [debug] au
```