Binary Exploitation aka Pwn Basic 補充篇

NTUSTISC

2021/5/26

whoami

- LJP / LJP-TW
- Pwn / Rev
- NTUST / ~~NCTU~~ / NYCU
- 10sec CTF Team

Outline

- TLS

TLS

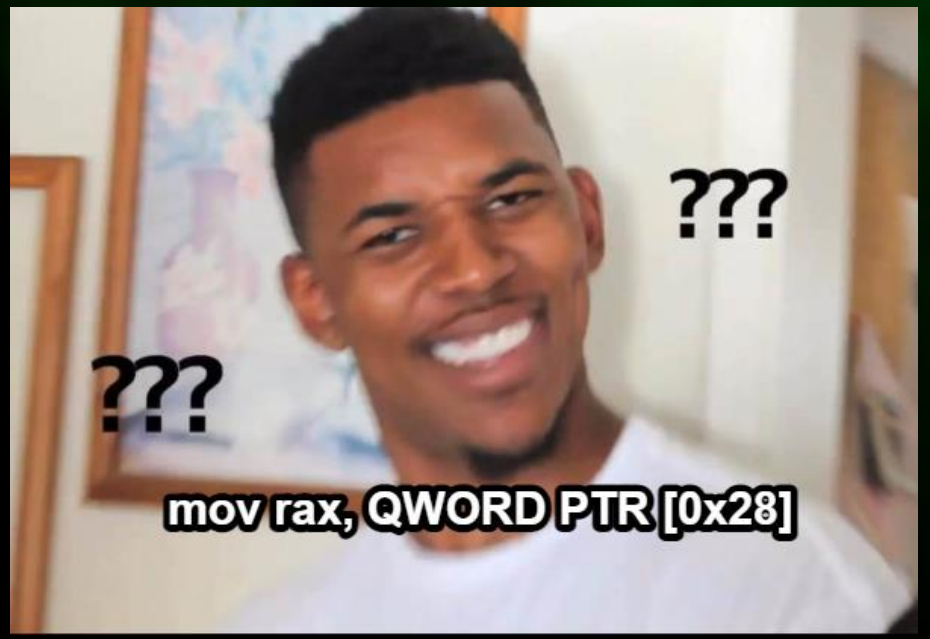
TLS

- TLS 全名 Thread-Local Storage
- Linux x64 使用 fs 暫存器記著 TLS 的位置
- Stack Canary 就是存在 TLS 中

```
Dump of assembler code for function main:
   0x000000000000124c <+0>:     endbr64
   0x0000000000001250 <+4>:     push   rbp
   0x0000000000001251 <+5>:     mov    rbp,rsp
   0x0000000000001254 <+8>:     sub    rsp,0x20
   0x0000000000001258 <+12>:    mov    rax,QWORD PTR fs:0x28
   0x0000000000001261 <+21>:    mov    QWORD PTR [rbp-0x8],rax
```

TLS

- fs 為 Segment Register
- 計算方式 reg:offset = ref + offset
- 這時候你用 gdb 想看一下 fs 等於多少卻發現 `$fs: 0×0000`
- 難道 Canary 從 [0+0x28] 拿來的??



TLS

- GDB 也是 Process, fs = 0 是指 GDB 自己的 fs
- 所以要怎麼拿到觀測中的 Process 的 fs?
- 呼叫 arch_prctl

```
gef➤  print (void)arch_prctl(0x1003, $rsp-8)
$1 = void
gef➤  x/xg $rsp-8
0x7fffffffe4c8: 0x00007ffff7fbc540
gef➤  x/8xg 0x00007ffff7fbc540
0x7ffff7fbc540: 0x00007ffff7fbc540      0x00007ffff7fbce80
0x7ffff7fbc550: 0x00007ffff7fbc540      0x0000000000000000
0x7ffff7fbc560: 0x0000000000000000      0x0e52fba2c545db00
0x7ffff7fbc570: 0x36aa29a7d2b974ac      0x0000000000000000
```

Ref: https://fasterthanli.me/series/making-our-own-executable-packer/part-13

TLS

- Pwngdb 有實作取得 TLS 的功能

```
gef➤  print (void)arch_prctl(0x1003, $rsp-8)
$2 = void
gef➤  x/xg $rsp-8
0x7fffffffe4c8: 0x00007ffff7fbc540
gef➤  tls
tls : 0x7ffff7fbc540
```

- 閱讀一下怎麼實作的, 發現其實一樣
  - https://github.com/scwuaptx/Pwngdb/blob/master/pwndbg/pwngdb.py#L77

TLS Demo