Zad 3.

środa, 12 kwietnia 2023 16:43

Zadanie 3. Zakładamy, że producent procesora nie dostarczył instrukcji **skoku pośredniego**. Rozważmy procedurę «switch_prob» z poprzedniej listy. Podaj metodę zastąpienia «jmpq *0x4006f8(,%rsi,8)» ciągiem innych instrukcji. Nie można używać **kodu samomodyfikującego się** (ang. *self-modyfing code*), ani dodatkowych rejestrów. Napisz kod w języku C, który wygeneruje instrukcję **pośredniego wywołania procedury**, np. «call *(%rdi,%rsi,8)», a następnie zaprezentuj go posługując się stroną godbolt[2]. Pokaż, że taką instrukcję też da się zastąpić, gdyby brakowało jej w zestawie instrukcji.

```
1  400590 <switch_prob>:
2  400590: 48 83 ef 3c             subq  $0x3c,%rsi
3  400594: 48 83 fe 05      .L01   cmpq  $0x5,%rsi
4  400598: 77 29                   ja    *0x4005c3
5  40059a: ff 24 f5 f8 06 40 00    jmpq  *0x4006f8(,%rsi,8)
6  4005a1: 48 8d 04 fd 00 00 00 00 lea   0x0(,%rdi,8),%rax
7  4005a9: c3                      retq
8  4005aa: 48 89 f8         .L4 :  movq  %rdi,%rax
9  4005ad: 48 c1 f8 03             sarq  $0x3,%rax
10 4005b1: c3                      retq
11 4005b2: 48 89 f8         .L2 :  movq  %rdi,%rax
12 4005b5: 48 c1 e0 04             shlq  $0x4,%rax
13 4005b9: 48 29 f8                subq  %rdi,%rax
14 4005bc: 48 89 c7                movq  %rax,%rdi
15 4005bf: 48 0f af ff      .L5    imulq %rdi,%rdi
16 4005c3: 48 8d 47 4b      .L3:   leaq  0x4b(%rdi),%rax
17 4005c7: c3                      retq
```

Zrzut pamięci przechowującej tablicę skoków:

```
18 (gdb) x/6gx 0x4006f8
19 0x4006f8: 0x4005a1
20 0x400700: 0x4005a1
21 0x400708: 0x4005b2
22 0x400710: 0x4005c3
23 0x400718: 0x4005aa
24 0x400720: 0x4005bf
```

0 1 2 3 4 5

a) jmpq *0x4006f8(, %rsi, 8)

↓

push 0x4006f8(, %rsi, 8)
ret

b)

```c
long test(long n, long (*func)(long)) {
    return func(n);
}
```

```
test:
        pushq   %rbp
        movq    %rsp, %rbp
        subq    $16, %rsp
        movq    %rdi, -8(%rbp)
        movq    %rsi, -16(%rbp)
        movq    -8(%rbp), %rax
        movq    -16(%rbp), %rdx
        movq    %rax, %rdi
        call    *%rdx
        leave
        ret
```

tutaj

c)

push .L1
push (%rdi, %rsi, 8)
ret
.L1 [...]