Lec03: Exploits Stack Buffer Overflow

Chengyu Song

Slides based on CS6265 taught by Prof. Taesoo Kim
Scoreboard
Administrative

- Survey1: need a break and review week?
- Survey2: need reference write-ups after deadline?
- Lab03: stack overflow is released!
- **Due**: Feb 2nd at midnight
Best Write-ups for Lab02

• Hang Zhang hzhan033@ucr.edu
Bomb Stats (Week 1)

- Bombs exploded 70 times in total \((146 \times -5 = -730 \text{ pts})\)
- in 7 phases, except 5,7,9
- 10 people exploded at least once! (so how many alive?)
  - Each lab: 22/11/10/05/00/03/00/03/00/01 people
  - Each lab: 71/40/12/15/00/04/00/03/00/01 times
Min shellcode

• 2000 bytes? 1000 bytes? 500 bytes? 100 bytes?

• ZERO!!
Discussion 0

• How different is the bomb binary this time?
Discussion 1

• How did you start exploring the "bomb" (no symbol)?
Discussion 2 (phase 1)

• What's going on the first phase?
Discussion 3 (phase 2)

- What's going on the second phase?
  - Did you find the main() function (i.e., dispatcher?)
Discussion 3 (obfuscation)

$ x /8i 0x004015f3

0x4015f3: sub rsp,0x8
0x4015f7: jmp 0x4015fa
0x4015f9: jmp 0x3f83e6
0x4015fe: dec DWORD PTR [rax-0x7d]
0x401601: (bad)
0x401602: or bl,al
0x401604: push rbx
0x401605: sub rsp,0x40
Discussion 3 (when tracing)

0x4015f3: sub rsp, 0x8
-> 0x4015f7: jmp 0x4015fa
| 0x4015f9: jmp 0x3f83e6
| 0x4015fe: dec DWORD PTR [rax-0x7d]
| 0x401601: (bad)
| 0x401602: or bl, al
+-> 0x4015fa: call 0x40156c
0x4015ff: add rsp, 0x8
0x401603: ret
0x401604: push rbx
Discussion 4 (phase 3)

- What's going on the third phase?
Discussion 4 (phase 3)

```c
int count = 0;
void progress_bar(int signo) {
    if (count != 0)
        printf("\b\b\b\b");
    printf("| %02d%%", count);
    count += 2;
}

phase() {
    signal(SIGTRAP, progress_bar);
    for (int i = 0; i < 50; i++) {
        ...
        __asm__ volatile("int3");
    }
}
```
Discussion 5 (phase 4)

1. What's going on the last phase? (nothing special!)
32/64 Shellcode

- restricted system calls
- int $0x80 vs. syscall (try \texttt{man syscall})
  - int $0x80: %ebx, %ecx, %edx, %esi, %edi, %ebp
  - syscall: %rdi, %rsi, %rdx, %r10, %r8, %r9
What's about poly shellcode?

1. What's your general idea?
Dispatching routine

+-------------------+
|                   |
[dispatcher][x86][x86_64]

e.g., 0x40 0x90
- x86    inc eax
- x86_64 REX + nop

x86   : [ * ][goto x86 shellcode]
x86-64: [nop][ * ][goto x86_64 shellcode]
arm    : [nop][nop][ * ][goto arm shellcode]
MIPS   : [nop][nop][nop][ * ][goto MIPS shellcode]
Discussion 6 (shellcode ascii/min)

- Wow, what are your tricks?
- How can be as small as zero byte?
Lab03: Stack overflow (due in two weeks)

• Finally! time to write real exploits (i.e., control hijacking)
• TONS of interesting challenges!
  • e.g., lack-of-four, frobnicated, upside-down ..
Today's Tutorial

- Example: hijacking crackme0x00!
- A template exploit code
- In-class tutorial
  - Extending the exploit template (python)
- PEDA
  - Follow the video
crackme0x00

$ objdump -d crackme0x00
...
8048414:   55 push %ebp
8048415:   89 e5 mov %esp,%ebp
8048417:   83 ec 28 sub %0x28,%esp

+--- ebp
top       v
[ ][fp][ra]
|<--- 0x28 ------->|
crackme0x00

$ objdump -d crackme0x00
...
8048448:   8d 45 e8   lea   -0x18(%ebp),%eax
804844b:   89 44 24 04  mov   %eax,0x4(%esp)
804844f:   c7 04 24 8c 85 04 08  movl  $0x804858c,%esp
8048456:   e8 d5 fe ff ff  call  8048330 <scanf@plt>

|<--  0x18-->|+---- ebp

top         v
[         [~~~> ] ][fp][ra]
|<----  0x28    -------->|
References

• IDA Demo
• Phrack #49-14