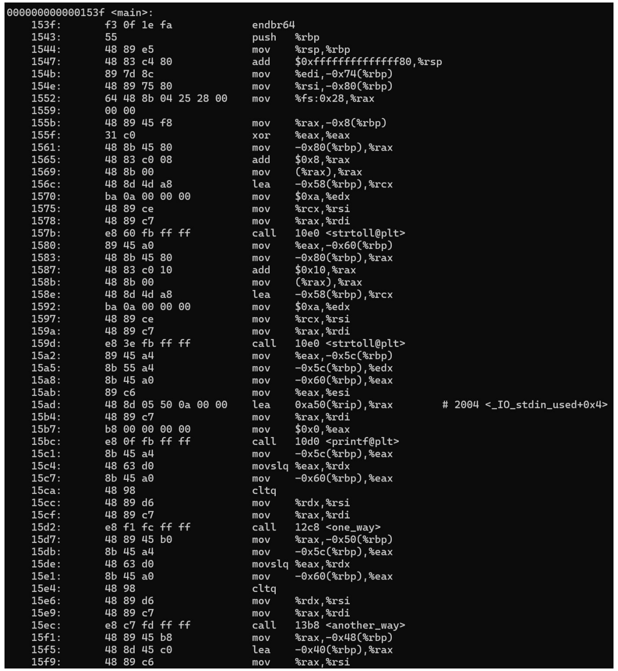
Recitation 9

Computer Architecture (section 1)

Dumping Info from Binary

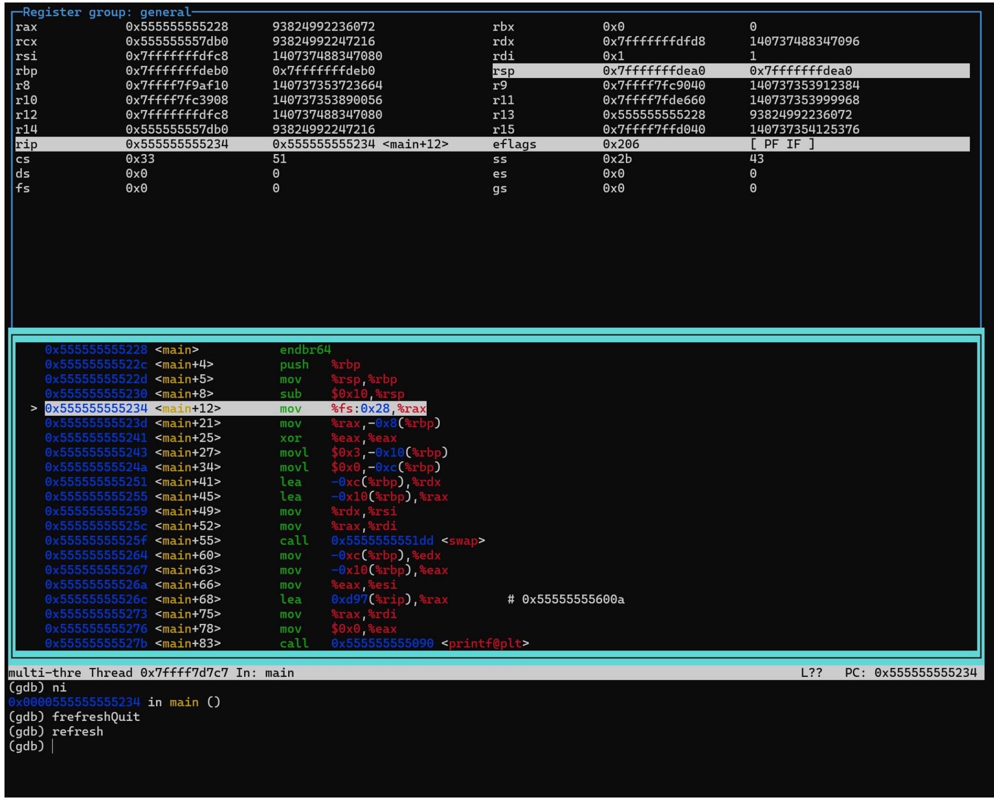
- Dump assembly
  - `objdump -d bomb > file.s`
- View strings
  - `strings bomb > strings.txt`



```
0000000000000153f <main>:
    153f:       f3 0f 1e fa             endbr64
    1543:       55                      push   %rbp
    1544:       48 89 e5                mov    %rsp,%rbp
    1547:       48 83 c4 80             add    $0xffffffffffffff80,%rsp
    154b:       89 7d 8c                mov    %edi,-0x74(%rbp)
    154e:       48 89 75 80             mov    %rsi,-0x80(%rbp)
    1552:       64 48 8b 04 25 28 00    mov    %fs:0x28,%rax
    1559:       00 00
    155b:       48 89 45 f8             mov    %rax,-0x8(%rbp)
    155f:       31 c0                   xor    %eax,%eax
    1561:       48 8b 45 80             mov    -0x80(%rbp),%rax
    1565:       48 83 c0 08             add    $0x8,%rax
    1569:       48 8b 00                mov    (%rax),%rax
    156c:       48 8d 4d a8             lea    -0x58(%rbp),%rcx
    1570:       ba 0a 00 00 00          mov    $0xa,%edx
    1575:       48 89 ce                mov    %rcx,%rsi
    1578:       48 89 c7                mov    %rax,%rdi
    157b:       e8 60 fb ff ff          call   10e0 <strtoll@plt>
    1580:       89 45 a0                mov    %eax,-0x60(%rbp)
    1583:       48 8b 45 80             mov    -0x80(%rbp),%rax
    1587:       48 83 c0 10             add    $0x10,%rax
    158b:       48 8b 00                mov    (%rax),%rax
    158e:       48 8d 4d a8             lea    -0x58(%rbp),%rcx
    1592:       ba 0a 00 00 00          mov    $0xa,%edx
    1597:       48 89 ce                mov    %rcx,%rsi
    159a:       48 89 c7                mov    %rax,%rdi
    159d:       e8 3e fb ff ff          call   10e0 <strtoll@plt>
    15a2:       89 45 a4                mov    %eax,-0x5c(%rbp)
    15a5:       8b 55 a4                mov    -0x5c(%rbp),%edx
    15a8:       8b 45 a0                mov    -0x60(%rbp),%eax
    15ab:       89 c6                   mov    %eax,%esi
    15ad:       48 8d 05 50 0a 00 00    lea    0xa50(%rip),%rax        # 2004 <_IO_stdin_used+0x4>
    15b4:       48 89 c7                mov    %rax,%rdi
    15b7:       b8 00 00 00 00          mov    $0x0,%eax
    15bc:       e8 0f fb ff ff          call   10d0 <printf@plt>
    15c1:       8b 45 a4                mov    -0x5c(%rbp),%eax
    15c4:       48 63 d0                movslq %eax,%rdx
    15c7:       8b 45 a0                mov    -0x60(%rbp),%eax
    15ca:       48 98                   cltq
    15cc:       48 89 d6                mov    %rdx,%rsi
    15cf:       48 89 c7                mov    %rax,%rdi
    15d2:       e8 f1 fc ff ff          call   12c8 <one_way>
    15d7:       48 89 45 b0             mov    %rax,-0x50(%rbp)
    15db:       8b 45 a4                mov    -0x5c(%rbp),%eax
    15de:       48 63 d0                movslq %eax,%rdx
    15e1:       8b 45 a0                mov    -0x60(%rbp),%eax
    15e4:       48 98                   cltq
    15e6:       48 89 d6                mov    %rdx,%rsi
    15e9:       48 89 c7                mov    %rax,%rdi
    15ec:       e8 c7 fd ff ff          call   13b8 <another_way>
    15f1:       48 89 45 b8             mov    %rax,-0x48(%rbp)
    15f5:       48 8d 45 c0             lea    -0x40(%rbp),%rax
    15f9:       48 89 c6                mov    %rax,%rsi
```

Debugging Assembly

- Assembly layout in GDB
  - Text user interface mode
    - ctrl-x + ctrl-a
    - `tui enable`
  - `layout asm`
  - `layout regs`
  - `focus cmd`
- Screen becomes glitched?
  - Redraw screen:
    - ctrl-L
    - `refresh`



```
┌Register group: general───────────────────────────────────────────────────────────────────────────────────┐
 rax            0x555555555228      93824992236072                 rbx            0x0                 0
 rcx            0x555555557db0      93824992247216                 rdx            0x7fffffffdfd8      140737488347096
 rsi            0x7fffffffdfc8      140737488347080                rdi            0x1                 1
 rbp            0x7fffffffdeb0      0x7fffffffdeb0                 rsp            0x7fffffffdea0      0x7fffffffdea0
 r8             0x7ffff7f9af10      140737353723664                r9             0x7ffff7fc9040      140737353912384
 r10            0x7ffff7fc3908      140737353890056                r11            0x7ffff7fde660      140737353999968
 r12            0x7fffffffdfc8      140737488347080                r13            0x555555555228      93824992236072
 r14            0x555555557db0      93824992247216                 r15            0x7ffff7ffd040      140737354125376
 rip            0x555555555234      0x555555555234 <main+12>       eflags         0x206               [ PF IF ]
 cs             0x33                51                             ss             0x2b                43
 ds             0x0                 0                              es             0x0                 0
 fs             0x0                 0                              gs             0x0                 0

     0x555555555228 <main>           endbr64
     0x55555555522c <main+4>         push   %rbp
     0x55555555522d <main+5>         mov    %rsp,%rbp
     0x555555555230 <main+8>         sub    $0x10,%rsp
   > 0x555555555234 <main+12>        mov    %fs:0x28,%rax
     0x55555555523d <main+21>        mov    %rax,-0x8(%rbp)
     0x555555555241 <main+25>        xor    %eax,%eax
     0x555555555243 <main+27>        movl   $0x3,-0x10(%rbp)
     0x55555555524a <main+34>        movl   $0x0,-0xc(%rbp)
     0x555555555251 <main+41>        lea    -0xc(%rbp),%rdx
     0x555555555255 <main+45>        lea    -0x10(%rbp),%rax
     0x555555555259 <main+49>        mov    %rdx,%rsi
     0x55555555525c <main+52>        mov    %rax,%rdi
     0x55555555525f <main+55>        call   0x5555555551dd <swap>
     0x555555555264 <main+60>        mov    -0xc(%rbp),%edx
     0x555555555267 <main+63>        mov    -0x10(%rbp),%eax
     0x55555555526a <main+66>        mov    %eax,%esi
     0x55555555526c <main+68>        lea    0xd97(%rip),%rax        # 0x55555555600a
     0x555555555273 <main+75>        mov    %rax,%rdi
     0x555555555276 <main+78>        mov    $0x0,%eax
     0x55555555527b <main+83>        call   0x555555555090 <printf@plt>

multi-thre Thread 0x7ffff7d7c7 In: main                                                   L??   PC: 0x555555555234
(gdb) ni
0x0000555555555234 in main ()
(gdb) frefreshQuit
(gdb) refresh
(gdb) 
```

Useful GDB Commands for Bomblab

- Step through each instruction
  - Into calls: `si`
  - Over calls: `ni`
- Write to memory or registers
  - `set *((int*)($rsp)) = 0x38f`
  - `set $rax = 5`
- Examine memory address
  - `x $rsp + 4`
  - `x/16d $rsp`
  - `x/s $rsp+4`
- Avoid bomb explosion
  - `break explode_bomb`

Bomblab Workflow

- Create the text file `~/.gdbinit`
  - Add initialization commands to it:

```
layout asm
layout regs
focus cmd
break explode_bomb
```

- On gdb startup
  - `source ~/.gdbinit`

PA4 Live Demo

In-class Bomb Defusal