Lab2: Bomblab

CSE4009: System Programming

Overview

- Download the handout from HConnect
- Place and extract the handout file
- Submit your solution file to LMS

Goal

- Learn how to read assembly code
- Learn how to use the tools necessary to deal with assembly code
  - gdb
  - objdump
  - strings

1. Check the handout file

- Download the handout file assigned to you

  $ git pull origin

```
wsul@splab2022012345:~/Projects/2022_cse4009_201220789$ git pull origin
Already up to date.
wsul@splab2022012345:~/Projects/2022_cse4009_201220789$
```

2. Extract handout file

- Can see the file in your project home directory and extract it

```
wsul@splab2022012345:~/Projects/2022_cse4009_201220789$ tar xvf bomblab.tar
./bomblab/
./bomblab/bomb
./bomblab/bomb.c
```

3. Check your files

- Can see two files on bomblab directory

```
wsul@splab2022012345:~/Projects/2022_cse4009_201220789$ cd bomblab/
wsul@splab2022012345:~/Projects/2022_cse4009_201220789/bomblab$ ll
total 44
drwxrwxr-x 2 wsul wsul  4096 Sep 29 13:11 ./
drwxrwxr-x 5 wsul wsul  4096 Sep 29 14:36 ../
-rwxrwxr-x 1 wsul wsul 31376 Sep 29 13:22 bomb*
-rwx------ 1 wsul wsul  4069 Sep 29 13:22 bomb.c*
wsul@splab2022012345:~/Projects/2022_cse4009_201220789/bomblab$ ./bomb
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
Hi

BOOM!!!
The bomb has blown up.
wsul@splab2022012345:~/Projects/2022_cse4009_201220789/bomblab$
```

4. The bomblab is...

- Consisted of 6 phases to defuse it
  - check it from bomb.c

woongsul — wsul@vbox: ~/Projects/labs/bomblab-handout — ssh -p 2222 wsul@localhost — 93×25

...jects/labs/bomblab-handout — ssh -p 2222 wsul@localhost | ...ads — woongsul@Woongs-MBP: ~/Downloads — -bash

```
  printf("Usage: %s [<input_file>]\n", argv[0]);
  exit(8);
    }

    /* Do all sorts of secret stuff that makes the bomb harder to defuse. */
    initialize_bomb();

    printf("Welcome to my fiendish little bomb. You have 6 phases with\n");
    printf("which to blow yourself up. Have a nice day!\n");

    /* Hmm...  Six phases must be more secure than one phase! */
    input = read_line();             /* Get input                   */
    phase_1(input);                  /* Run the phase               */
    phase_defused();                 /* Drat!  They figured it out!
              * Let me know how they did it. */
    printf("Phase 1 defused. How about the next one?\n");

    /* The second phase is harder.  No one will ever figure out
     * how to defuse this... */
    input = read_line();
    phase_2(input);
    phase_defused();
    printf("That's number 2.  Keep going!\n");

                                                          73,2          67%
```

4. The bomblab is...

- Consisted of 6 phases to defuse it
  - check it from bomb.c
  - you have to make the right answer for each phase

5. How to defuse it?

- Let’s check the example
  - gdb shows that "strings_not_equals" function checks for the answer

```
woongsul — wsul@vbox: ~/Projects/labs/bomblab-handout — ssh -p 2222 wsul@localhost — 93×25
...jects/labs/bomblab-handout — ssh -p 2222 wsul@localhost    ...ads — woongsul@Woongs-MBP: ~/Downloads — -bash
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from bomb...done.
(gdb) b phase_1
Breakpoint 1 at 0x1204
(gdb) r
Starting program: /home/wsul/Projects/labs/bomblab-handout/bomb
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
1111

Breakpoint 1, 0x0000555555555204 in phase_1 ()
(gdb) disassemble
Dump of assembler code for function phase_1:
=> 0x0000555555555204 <+0>:     sub    $0x8,%rsp
   0x0000555555555208 <+4>:     lea    0x1621(%rip),%rsi        # 0x555555556830
   0x000055555555520f <+11>:    callq  0x555555555737 <strings_not_equal>
   0x0000555555555214 <+16>:    test   %eax,%eax
   0x0000555555555216 <+18>:    jne    0x55555555521d <phase_1+25>
   0x0000555555555218 <+20>:    add    $0x8,%rsp
   0x000055555555521c <+24>:    retq
   0x000055555555521d <+25>:    callq  0x555555555843 <explode_bomb>
   0x0000555555555222 <+30>:    jmp    0x555555555218 <phase_1+20>
End of assembler dump.
(gdb)
```

5. How to defuse it?

- Let’s check the example
  - gdb shows that "strings_not_equals" function checks for the answer

```
woongsul — wsul@vbox: ~/Projects/labs/bomblab-handout — ssh -p 2222 wsul@localhost — 93×25
...jects/labs/bomblab-handout — ssh -p 2222 wsul@localhost    ...ads — woongsul@Woongs-MBP: ~/Downloads — -bash
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from bomb...done.
(gdb) b phase_1
Breakpoint 1 at 0x1204
(gdb) r
Starting program: /home/wsul/Projects/labs/bomblab-handout/bomb
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
1111

Breakpoint 1, 0x0000555555555204 in phase_1 ()
(gdb) disassemble
Dump of assembler code for function phase_1:
=> 0x0000555555555204 <+0>:     sub    $0x8,%rsp
   0x0000555555555208 <+4>:     lea    0x1621(%rip),%rsi        # 0x555555556830
   0x000055555555520f <+11>:    callq  0x555555555737 <strings_not_equal>
   0x0000555555555214 <+16>:    test   %eax,%eax
   0x0000555555555216 <+18>:    jne    0x55555555521d <phase_1+25>
   0x0000555555555218 <+20>:    add    $0x8,%rsp
   0x000055555555521c <+24>:    retq
   0x000055555555521d <+25>:    callq  0x555555555843 <explode_bomb>
   0x0000555555555222 <+30>:    jmp    0x555555555218 <phase_1+20>
End of assembler dump.
(gdb)
```

5. How to defuse it?

- Let’s check the example
  - gdb shows that "strings_not_equals" function checks for the answer

woongsul — wsul@vbox: ~/Projects/labs/bomblab-handout — ssh -p 2222 wsul@localhost — 93×26

...jects/labs/bomblab-handout — ssh -p 2222 wsul@localhost | ...ads — woongsul@Woongs-MBP: ~/Downloads — -bash

```
Dump of assembler code for function strings_not_equal:
=> 0x0000555555555737 <+0>:     push   %r12
   0x0000555555555739 <+2>:     push   %rbp
   0x000055555555573a <+3>:     push   %rbx
   0x000055555555573b <+4>:     mov    %rdi,%rbx
   0x000055555555573e <+7>:     mov    %rsi,%rbp
   0x0000555555555741 <+10>:    callq  0x55555555571a <string_length>
   0x0000555555555746 <+15>:    mov    %eax,%r12d
   0x0000555555555749 <+18>:    mov    %rbp,%rdi
   0x000055555555574c <+21>:    callq  0x55555555571a <string_length>
   0x0000555555555751 <+26>:    mov    $0x1,%edx
   0x0000555555555756 <+31>:    cmp    %eax,%r12d
   0x0000555555555759 <+34>:    je     0x555555555762 <strings_not_equal+43>
   0x000055555555575b <+36>:    mov    %edx,%eax
   0x000055555555575d <+38>:    pop    %rbx
   0x000055555555575e <+39>:    pop    %rbp
   0x000055555555575f <+40>:    pop    %r12
   0x0000555555555761 <+42>:    retq
   0x0000555555555762 <+43>:    movzbl (%rbx),%eax
   0x0000555555555765 <+46>:    test   %al,%al
   0x0000555555555767 <+48>:    je     0x555555555790 <strings_not_equal+89>
   0x0000555555555769 <+50>:    cmp    0x0(%rbp),%al
   0x000055555555576c <+53>:    jne    0x555555555797 <strings_not_equal+96>
   0x000055555555576e <+55>:    add    $0x1,%rbx
   0x0000555555555772 <+59>:    add    $0x1,%rbp
---Type <return> to continue, or q <return> to quit---
```

5. How to defuse it?

- Let’s check the example
  - gdb shows that "strings_not_equals" function checks for the answer
  - You can guess two strings to compare are stored in rdi, rsi registers

woongsul — wsul@vbox: ~/Projects/labs/bomblab-handout — ssh -p 2222 wsul@localhost — 93×26

...jects/labs/bomblab-handout — ssh -p 2222 wsul@localhost | ...ads — woongsul@Woongs-MBP: ~/Downloads — -bash

```
Dump of assembler code for function strings_not_equal:
=> 0x0000555555555737 <+0>:     push   %r12
   0x0000555555555739 <+2>:     push   %rbp
   0x000055555555573a <+3>:     push   %rbx
   0x000055555555573b <+4>:     mov    %rdi,%rbx
   0x000055555555573e <+7>:     mov    %rsi,%rbp
   0x0000555555555741 <+10>:    callq  0x55555555571a <string_length>
   0x0000555555555746 <+15>:    mov    %eax,%r12d
   0x0000555555555749 <+18>:    mov    %rbp,%rdi
   0x000055555555574c <+21>:    callq  0x55555555571a <string_length>
   0x0000555555555751 <+26>:    mov    $0x1,%edx
   0x0000555555555756 <+31>:    cmp    %eax,%r12d
   0x0000555555555759 <+34>:    je     0x555555555762 <strings_not_equal+43>
   0x000055555555575b <+36>:    mov    %edx,%eax
   0x000055555555575d <+38>:    pop    %rbx
   0x000055555555575e <+39>:    pop    %rbp
   0x000055555555575f <+40>:    pop    %r12
   0x0000555555555761 <+42>:    retq
   0x0000555555555762 <+43>:    movzbl (%rbx),%eax
   0x0000555555555765 <+46>:    test   %al,%al
   0x0000555555555767 <+48>:    je     0x555555555790 <strings_not_equal+89>
   0x0000555555555769 <+50>:    cmp    0x0(%rbp),%al
   0x000055555555576c <+53>:    jne    0x555555555797 <strings_not_equal+96>
   0x000055555555576e <+55>:    add    $0x1,%rbx
   0x0000555555555772 <+59>:    add    $0x1,%rbp
---Type <return> to continue, or q <return> to quit---
```

5. How to defuse it?

- Let’s check the example
  - gdb shows that "strings_not_equals" function checks for the answer
  - You can guess two strings to compare are stored in rdi, rsi registers
  - Let’s check what is stored in each register

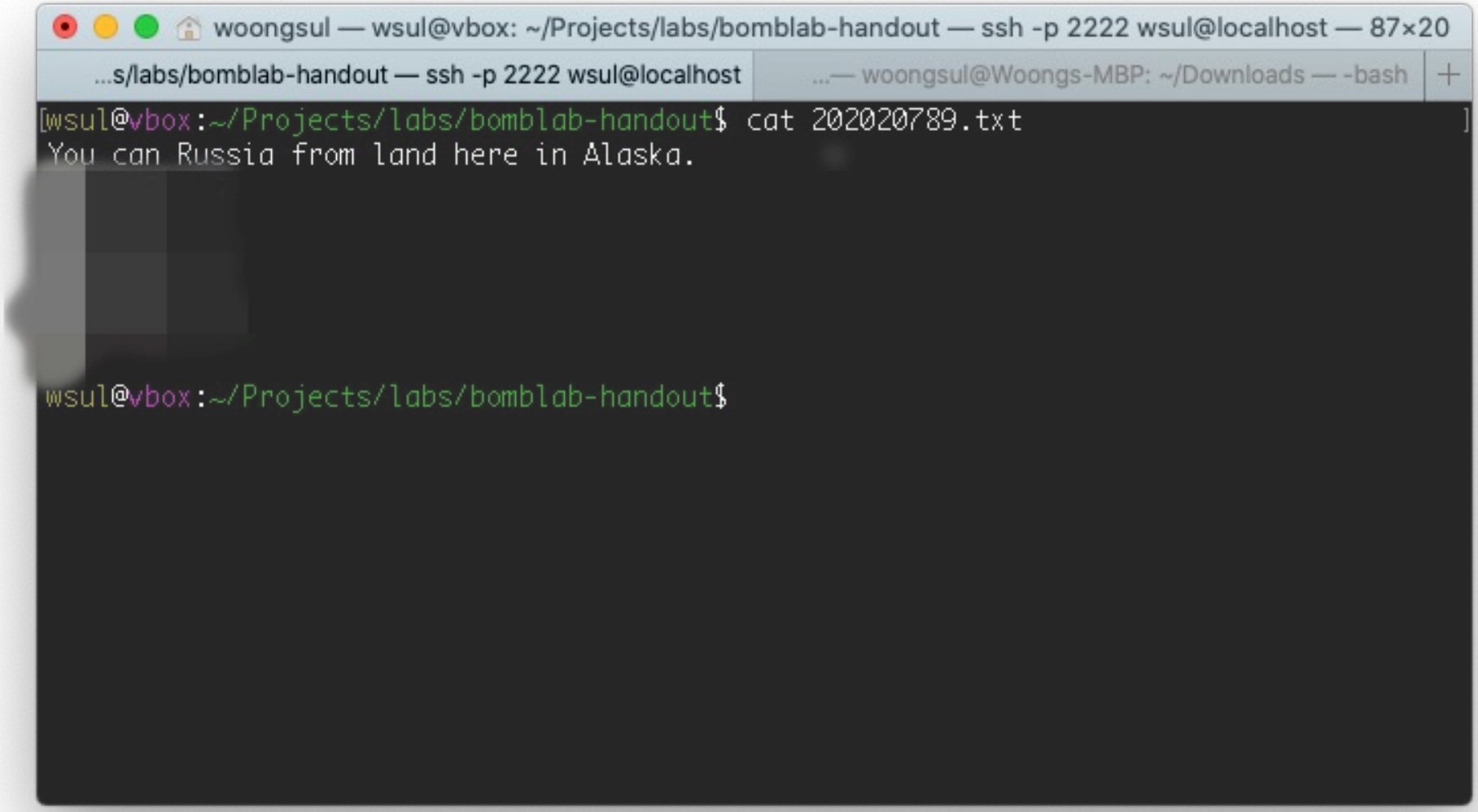
```
woongsul — wsul@vbox: ~/Projects/labs/bomblab-handout — ssh -p 2222 wsul@localhost — 93×26
...jects/labs/bomblab-handout — ssh -p 2222 wsul@localhost   ...ads — woongsul@Woongs-MBP: ~/Downloads — -bash
   0x000055555555574c <+21>:    callq  0x55555555571a <string_length>
   0x0000555555555751 <+26>:    mov    $0x1,%edx
   0x0000555555555756 <+31>:    cmp    %eax,%r12d
   0x0000555555555759 <+34>:    je     0x555555555762 <strings_not_equal+43>
   0x000055555555575b <+36>:    mov    %edx,%eax
   0x000055555555575d <+38>:    pop    %rbx
   0x000055555555575e <+39>:    pop    %rbp
   0x000055555555575f <+40>:    pop    %r12
   0x0000555555555761 <+42>:    retq
   0x0000555555555762 <+43>:    movzbl (%rbx),%eax
   0x0000555555555765 <+46>:    test   %al,%al
   0x0000555555555767 <+48>:    je     0x555555555790 <strings_not_equal+89>
   0x0000555555555769 <+50>:    cmp    0x0(%rbp),%al
   0x000055555555576c <+53>:    jne    0x555555555797 <strings_not_equal+96>
   0x000055555555576e <+55>:    add    $0x1,%rbx
   0x0000555555555772 <+59>:    add    $0x1,%rbp
[---Type <return> to continue, or q <return> to quit---q                    ]
Quit
[(gdb) info registers rdi rsi                                               ]
rdi            0x555555758ac0   93824994347712
rsi            0x555555556830   93824992241712
[(gdb) x/s 0x555555758ac0                                                   ]
0x555555758ac0 <input_strings>: "1111"   I made this string...
[(gdb) x/s 0x555555556830                                                   ]
0x555555556830: "You can Russia from land here in Alaska."  WHOA?????
(gdb)
```

5. How to defuse it?

- Let’s check the example
  - gdb shows that "strings_not_equals" function checks for the answer
  - You can guess two strings to compare are stored in rdi, rsi registers
  - Let’s check what is stored in each register
  - You have 3 more phase to defuse this bomb
  - Your strong friends..
    - gdb
      - you can examine commands in the previous example
    - objdump
      - -t : print out bomb’s symbol table
      - -d : print out disassemble code (or you can use disassemble in gdb)
    - strings
      - printout all printable strings in your bomb

6. Submission

- You are supposed to submit the answer file that lists each answer
  - Your answer file might be....



Good Luck!