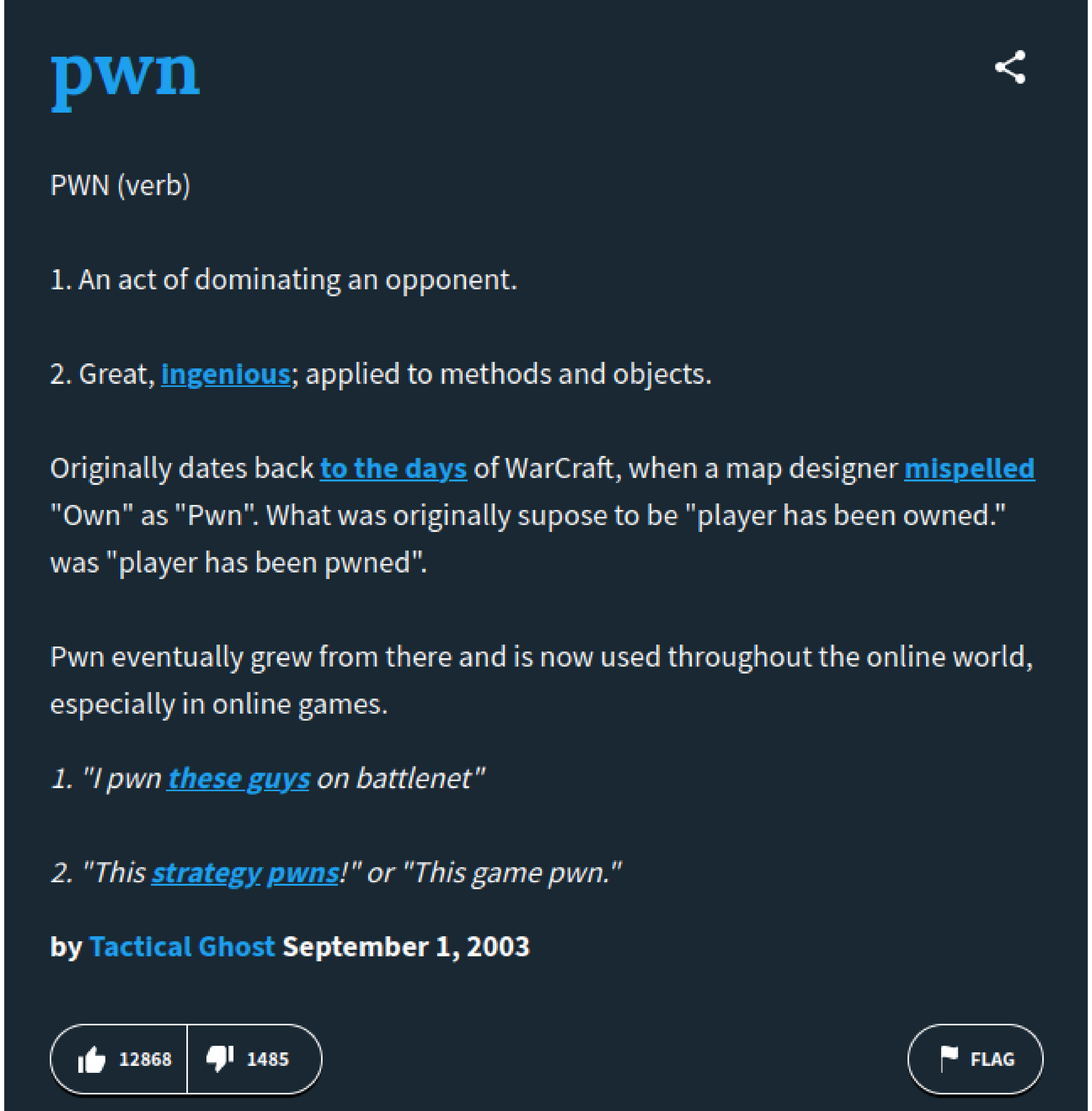
Introduction à l'exploitation de vulnérabilités

Sommaire

PWN ? OWN ?



pwn

PWN (verb)

1. An act of dominating an opponent.

2. Great, ingenious; applied to methods and objects.

Originally dates back to the days of WarCraft, when a map designer mispelled "Own" as "Pwn". What was originally supose to be "player has been owned." was "player has been pwned".

Pwn eventually grew from there and is now used throughout the online world, especially in online games.

1. "I pwn these guys on battlenet"

2. "This strategy pwns!" or "This game pwn."

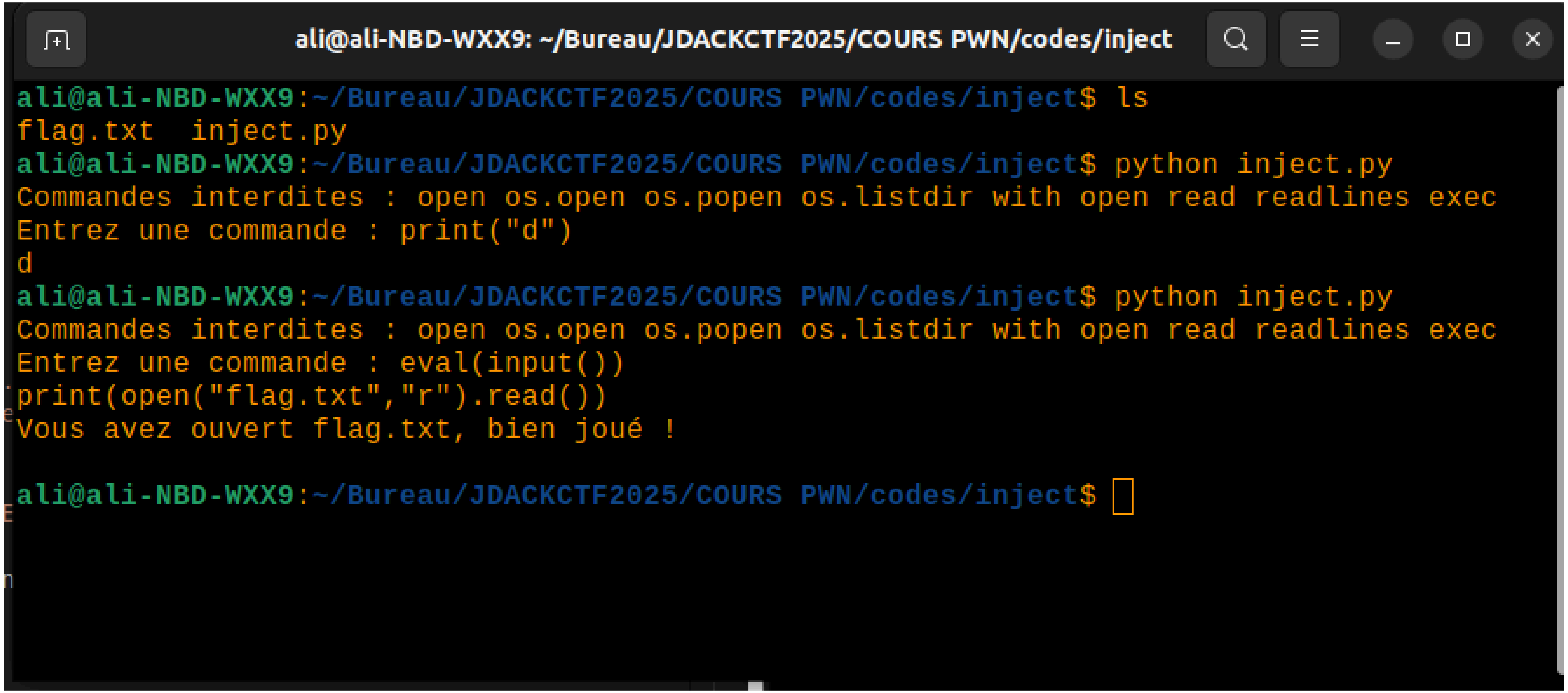
by Tactical Ghost September 1, 2003

Exploitation et sécurité

1. **Type de vulnérabilités**
2. **Mesure de sécurités**

Injection de commande

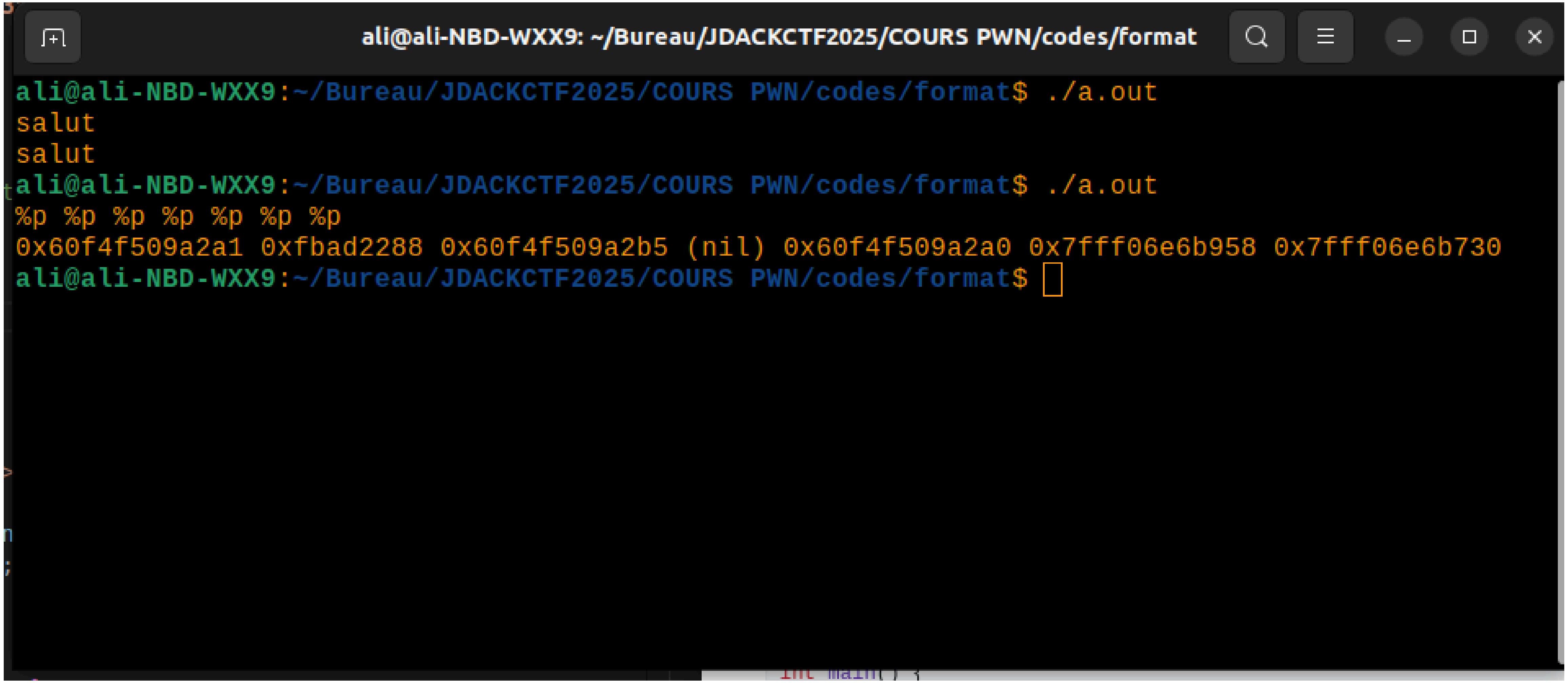
```python
import os

banned = "open os.open os.popen os.listdir with open read readlines exec"
print(f"Commandes interdites : {banned}")

command = input("Entrez une commande : ")

if command in banned:
    print("Erreur : Commande interdite !")
else:
    eval(command)
```


ali@ali-NBD-WXX9: ~/Bureau/JDACKCTF2025/COURS PWN/codes/inject

```
ali@ali-NBD-WXX9:~/Bureau/JDACKCTF2025/COURS PWN/codes/inject$ ls
flag.txt  inject.py
ali@ali-NBD-WXX9:~/Bureau/JDACKCTF2025/COURS PWN/codes/inject$ python inject.py
Commandes interdites : open os.open os.popen os.listdir with open read readlines exec
Entrez une commande : print("d")
d
ali@ali-NBD-WXX9:~/Bureau/JDACKCTF2025/COURS PWN/codes/inject$ python inject.py
Commandes interdites : open os.open os.popen os.listdir with open read readlines exec
Entrez une commande : eval(input())
print(open("flag.txt","r").read())
Vous avez ouvert flag.txt, bien joué !

ali@ali-NBD-WXX9:~/Bureau/JDACKCTF2025/COURS PWN/codes/inject$
```

Attaque par chaîne de format

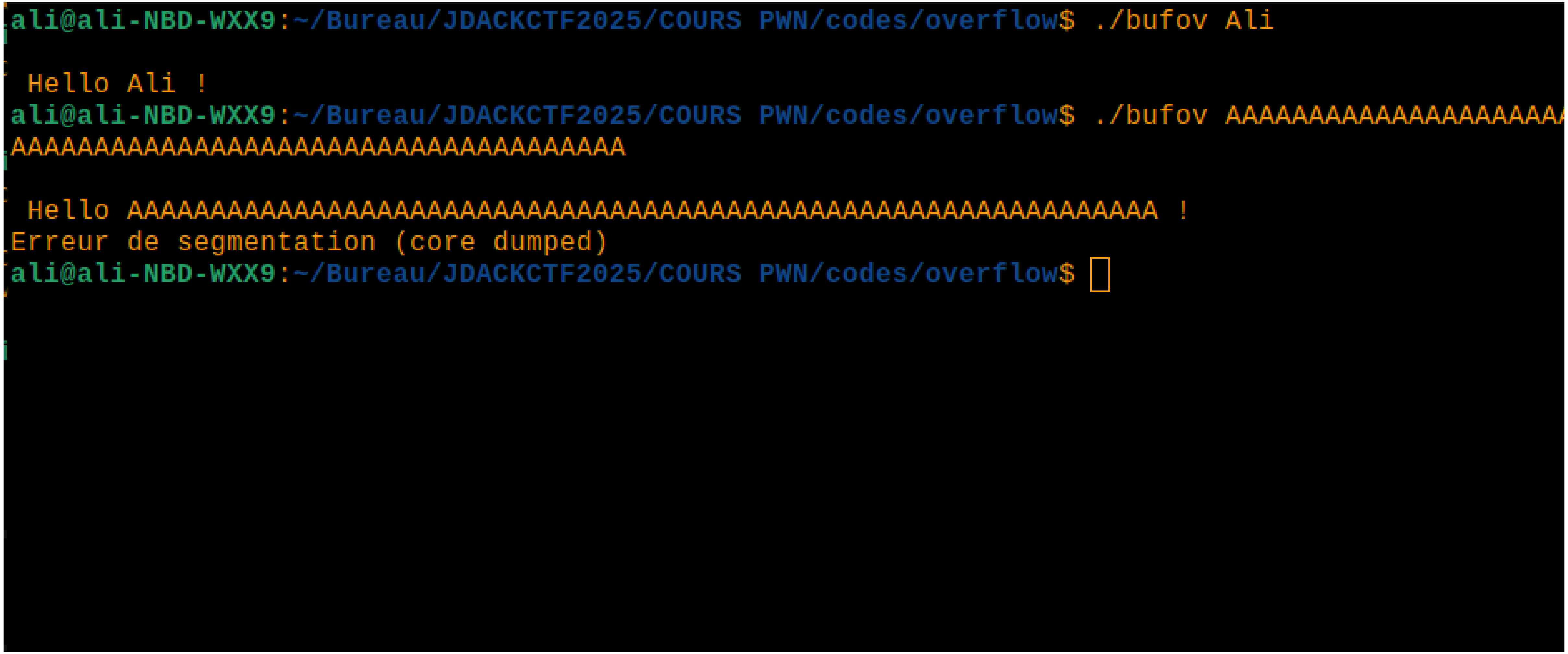
```c
#include <stdio.h>

void vuln(char *input) {
    printf(input);
}

int main() {
    char buffer[256];
    fgets(buffer, sizeof(buffer), stdin);
    vuln(buffer);
    return 0;
}
```


ali@ali-NBD-WXX9: ~/Bureau/JDACKCTF2025/COURS PWN/codes/format

```
ali@ali-NBD-WXX9:~/Bureau/JDACKCTF2025/COURS PWN/codes/format$ ./a.out
salut
salut
ali@ali-NBD-WXX9:~/Bureau/JDACKCTF2025/COURS PWN/codes/format$ ./a.out
%p %p %p %p %p %p %p
0x60f4f509a2a1 0xfbad2288 0x60f4f509a2b5 (nil) 0x60f4f509a2a0 0x7fff06e6b958 0x7fff06e6b730
ali@ali-NBD-WXX9:~/Bureau/JDACKCTF2025/COURS PWN/codes/format$
```

Buffer Overflow

```c
#include <stdio.h>
#include <string.h>

void vuln(char *arg) {
    char buffer[20];
    strcpy(buffer,arg);
    printf("%s\n",buffer);
}

int main(int argc, char **argv) {
    vuln(argv[1]);
    return 0;
}
```



```
ali@ali-NBD-WXX9:~/Bureau/JDACKCTF2025/COURS PWN/codes/overflow$ ./bufov Ali

 Hello Ali !
ali@ali-NBD-WXX9:~/Bureau/JDACKCTF2025/COURS PWN/codes/overflow$ ./bufov AAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

 Hello AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA !
Erreur de segmentation (core dumped)
ali@ali-NBD-WXX9:~/Bureau/JDACKCTF2025/COURS PWN/codes/overflow$
```

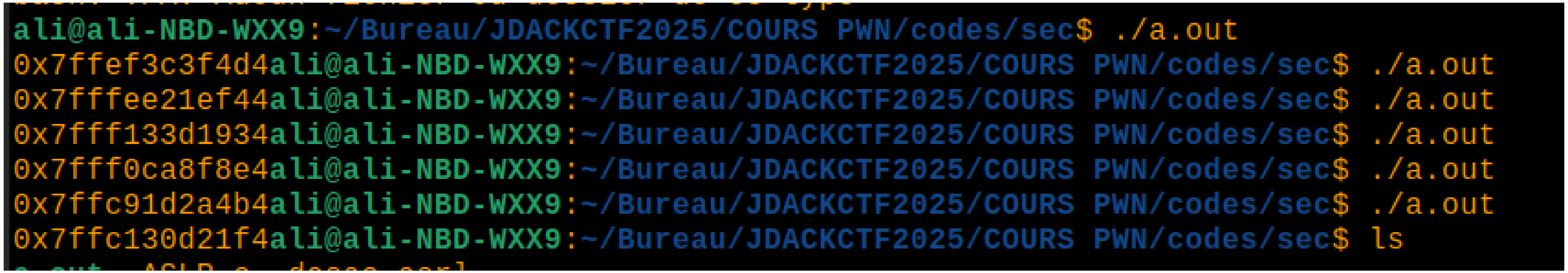
Mesure de sécurité

1. **ASLR / PIE / RELRO**
2. **Canary**
3. **NX**

ASLR exemple :

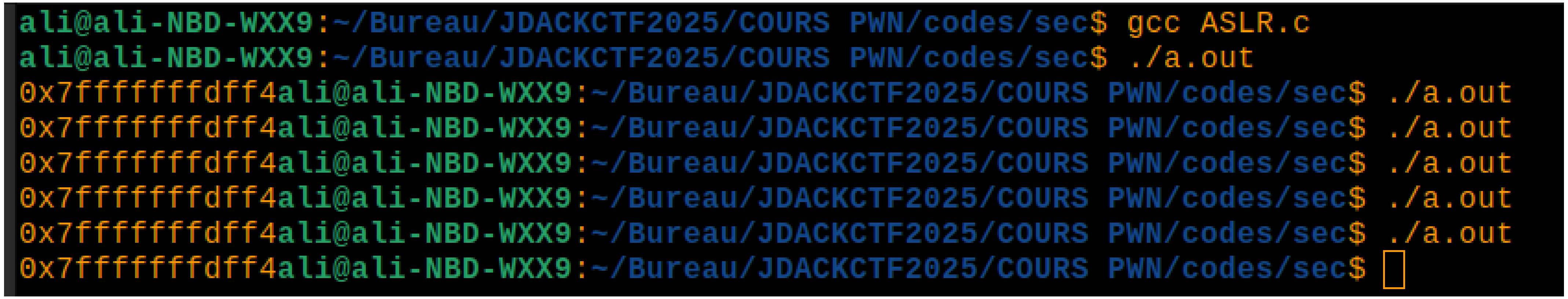
```
#include <stdlib.h>
#include <stdio.h>

int main() {
    int a = 10;
    printf("%p",&a);
    return EXIT_SUCCESS;
}
```

Éxecution avec ASLR



```
ali@ali-NBD-WXX9:~/Bureau/JDACKCTF2025/COURS PWN/codes/sec$ ./a.out
0x7ffef3c3f4d4ali@ali-NBD-WXX9:~/Bureau/JDACKCTF2025/COURS PWN/codes/sec$ ./a.out
0x7fffee21ef44ali@ali-NBD-WXX9:~/Bureau/JDACKCTF2025/COURS PWN/codes/sec$ ./a.out
0x7fff133d1934ali@ali-NBD-WXX9:~/Bureau/JDACKCTF2025/COURS PWN/codes/sec$ ./a.out
0x7fff0ca8f8e4ali@ali-NBD-WXX9:~/Bureau/JDACKCTF2025/COURS PWN/codes/sec$ ./a.out
0x7ffc91d2a4b4ali@ali-NBD-WXX9:~/Bureau/JDACKCTF2025/COURS PWN/codes/sec$ ./a.out
0x7ffc130d21f4ali@ali-NBD-WXX9:~/Bureau/JDACKCTF2025/COURS PWN/codes/sec$ ls
```

Éxecution sans ASLR



```
ali@ali-NBD-WXX9:~/Bureau/JDACKCTF2025/COURS PWN/codes/sec$ gcc ASLR.c
ali@ali-NBD-WXX9:~/Bureau/JDACKCTF2025/COURS PWN/codes/sec$ ./a.out
0x7fffffffdff4ali@ali-NBD-WXX9:~/Bureau/JDACKCTF2025/COURS PWN/codes/sec$ ./a.out
0x7fffffffdff4ali@ali-NBD-WXX9:~/Bureau/JDACKCTF2025/COURS PWN/codes/sec$ ./a.out
0x7fffffffdff4ali@ali-NBD-WXX9:~/Bureau/JDACKCTF2025/COURS PWN/codes/sec$ ./a.out
0x7fffffffdff4ali@ali-NBD-WXX9:~/Bureau/JDACKCTF2025/COURS PWN/codes/sec$ ./a.out
0x7fffffffdff4ali@ali-NBD-WXX9:~/Bureau/JDACKCTF2025/COURS PWN/codes/sec$ ./a.out
0x7fffffffdff4ali@ali-NBD-WXX9:~/Bureau/JDACKCTF2025/COURS PWN/codes/sec$
```

Mesure de sécurité

1. **ASLR / PIE / RELRO**
2. **Canary**
3. **NX**

Quelques outils..

1. **Checksec**
2. **GDB**
3. **Pwntools**

Aïe..

```c
void fonction1() {
    int a = 5, b = 0;
    printf("Résultat de la division : %d\n", a/b);
}
void fonction2() {
    int x = 10;
    fonction1();
    printf("Fin de fonction2, x = %d\n", x);
}

int main() {
    fonction2();
    return 0;
}
```

Communication avec pwntools

```python
from pwn import *
io = process('sh')
io.sendline(b'echo Hello, world')
response = io.recvline()
print(response.decode())
```

```python
from pwn import *
io = process('./programme_shell')
io.interactive()
```

Place à la pratique !