Snake

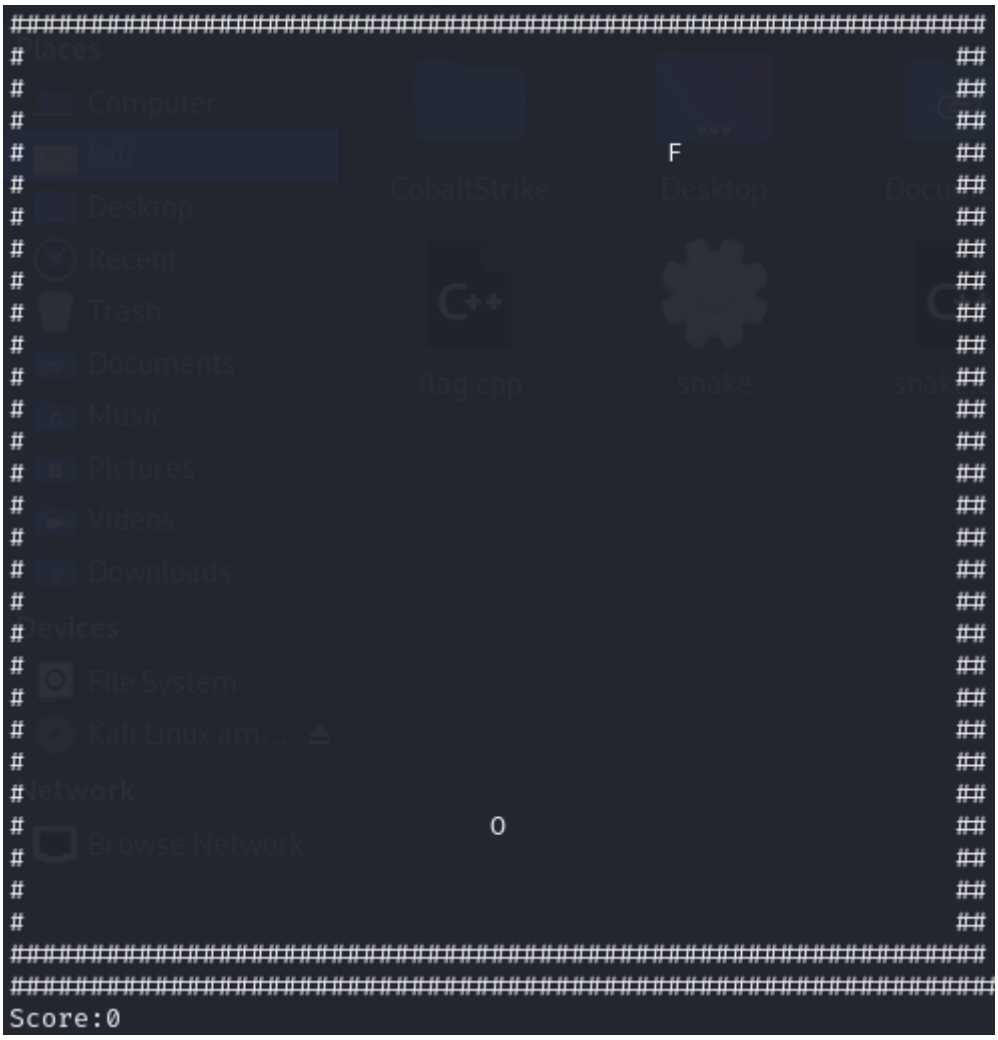
Des informations classifiées secret défense ont été cachées, il vous faudra obtenir plus de 100000 points pour accéder au code secret.

Write-up

Exécution du code :

```
./snake
```

Le jeu apparaît, nous pouvons nous déplacer avec zqsd. Chaque fruit donne 10 points. La grille du jeu fait 30x30, multiplier par 10 pour chaque fruit, on obtient un score maximal de 9000 points.



Lancement du binaire avec radare 2 :

```
radare2 -AAAA -d snake
```

Rien d'intéressant avec afl.

```
s main =
r_num_calc error: ( ')' expected) in (1000l> >(std::chrono::duration<long,
std::ratio<1l, 1000l> > const&))
            ; DATA XREF from entry0 @ 0x5583cd54f184
┌ 106: int main (int argc, char **argv, char **envp);
│           ; var int64_t var_10h @ rbp-0x10
│           ; var int64_t var_4h @ rbp-0x4
│           0x5583cd54fb08      55              push rbp
│           0x5583cd54fb09      4889e5          mov rbp, rsp
│           0x5583cd54fb0c      4883ec10        sub rsp, 0x10
│           0x5583cd54fb10      bf00000000      mov edi, 0
│           0x5583cd54fb15      e8a6f5ffff      call sym.imp.time       ; time_t
time(time_t *timer)
│           0x5583cd54fb1a      89c7            mov edi, eax
│           0x5583cd54fb1c      e8aff5ffff      call sym.imp.srand      ; void
srand(int seed)
│           0x5583cd54fb21      e833f7ffff      call sym Setup()        ;
sym.Setup__
│       ┌─< 0x5583cd54fb26      eb35            jmp 0x5583cd54fb5d
│      ┌──> 0x5583cd54fb28      e868faffff      call sym Draw()         ; sym.Draw__
│      ¦¦   0x5583cd54fb2d      e8a5f9ffff      call sym Input()        ;
sym.Input__
│      ¦¦   0x5583cd54fb32      e8fefcffff      call sym Logic()        ;
sym.Logic__
│      ¦¦   0x5583cd54fb37      c745fc640000.   mov dword [var_4h], 0x64 ; 'd' ; 100
│      ¦¦   0x5583cd54fb3e      488d55fc        lea rdx, [var_4h]
│      ¦¦   0x5583cd54fb42      488d45f0        lea rax, [var_10h]
│      ¦¦   0x5583cd54fb46      4889d6          mov rsi, rdx
│      ¦¦   0x5583cd54fb49      4889c7          mov rdi, rax
│      ¦¦   0x5583cd54fb4c      e865040000      call sym std::chrono::duration<long,
std::ratio<1l, 1000l> >::duration<int, void>(int const&) ;
sym.std::chrono::duration_long__std::ratio_1l__1000l___::duration_int__void__int_co
nst_
│      ¦¦   0x5583cd54fb51      488d45f0        lea rax, [var_10h]
│      ¦¦   0x5583cd54fb55      4889c7          mov rdi, rax
│      ¦¦   0x5583cd54fb58      e878040000      call sym void
std::this_thread::sleep_for<long, std::ratio<1l, 1000l> >
(std::chrono::duration<long, std::ratio<1l, 1000l> > const&) ;
sym.void_std::this_thread::sleep_for_long__std::ratio_1l__1000l____std::chrono::dur
ation_long__std::ratio_1l__1000l____const_
│      ¦¦   ; CODE XREF from main @ 0x5583cd54fb26
```

```
|      ┆└─> 0x5583cd54fb5d      0fb605d43600.  movzx eax, byte [0x5583cd553238] ;
[0x5583cd553238:1]=0
|      ┆    0x5583cd54fb64      83f001         xor eax, 1
|      ┆    0x5583cd54fb67      84c0           test al, al
|      └──< 0x5583cd54fb69      75bd           jne 0x5583cd54fb28
|           0x5583cd54fb6b      b800000000     mov eax, 0
|           0x5583cd54fb70      c9             leave
└           0x5583cd54fb71      c3             ret
```

Étant donné que nous devons agir sur le "score" nous Pouvons-nous attendre à une variable. Nous pouvons la retrouver avec la commande is :

```
is
163 0x00001fd5 0x5583cd54ffd5 WEAK   FUNC   193      void
std::this_thread::sleep_for<long, std::ratio<1l, 1000l> >
(std::chrono::duration<long, std::ratio<1l, 1000l> > const&)
164 ---------- 0x00005234     GLOBAL OBJ    4        score
165 0x00001cca 0x5583cd54fcca WEAK   FUNC   17       std::chrono::duration<long,
std::ratio<1l, 1l> >::count() const
```

En partant du principe qu'il s'agirait d'un objet, nous pouvions retrouver la variable avec la commande :

```
is | grep OBJ
28  ---------- 0x000050c0     GLOBAL OBJ    8        stdin
29  ---------- 0x00005100     GLOBAL OBJ    272      std::cout
2   0x0000037c 0x5583cd54e37c LOCAL  OBJ    32       __abi_tag
7   ---------- 0x00005210     LOCAL  OBJ    1        completed.0
8   0x00004dc0 0x5583cd552dc0 LOCAL  OBJ    0
__do_global_dtors_aux_fini_array_entry
10  0x00004db0 0x5583cd552db0 LOCAL  OBJ    0        __frame_dummy_init_array_entry
13  0x000030da 0x5583cd5510da LOCAL  OBJ    1
std::__detail::__integer_to_chars_is_unsigned<unsigned int>
14  0x000030db 0x5583cd5510db LOCAL  OBJ    1
std::__detail::__integer_to_chars_is_unsigned<unsigned long>
15  0x000030dc 0x5583cd5510dc LOCAL  OBJ    1
std::__detail::__integer_to_chars_is_unsigned<unsigned long long>
16  0x000030dd 0x5583cd5510dd LOCAL  OBJ    1
std::__is_ratio_v<std::ratio<1l, 1000000000l> >
17  0x000030de 0x5583cd5510de LOCAL  OBJ    1
```

```
std::__is_ratio_v<std::ratio<1l, 1l> >
18  0x000030df 0x5583cd5510df LOCAL  OBJ    1
std::__is_ratio_v<std::ratio<1000000000l, 1l> >
19  0x000030e0 0x5583cd5510e0 LOCAL  OBJ    1
std::__is_ratio_v<std::ratio<1l, 1000l> >
20  0x000030e1 0x5583cd5510e1 LOCAL  OBJ    1
std::__is_ratio_v<std::ratio<1000l, 1l> >
23  0x00003f94 0x5583cd551f94 LOCAL  OBJ    0        __FRAME_END__
26  0x00004dc8 0x5583cd552dc8 LOCAL  OBJ    0        _DYNAMIC
27  0x00004fe8 0x5583cd552fe8 LOCAL  OBJ    0        _GLOBAL_OFFSET_TABLE_
30  0x000050ac 0x5583cd5530ac GLOBAL OBJ    4        height
35  ---------- 0x00005230     GLOBAL OBJ    4        fruitY
41  ---------- 0x00005238     GLOBAL OBJ    1        gameOver
49  0x00003000 0x5583cd551000 GLOBAL OBJ    4        _IO_stdin_used
61  0x000050a0 0x5583cd5530a0 GLOBAL OBJ    0        __dso_handle
65  0x000050b0 0x5583cd5530b0 WEAK   OBJ    8        DW.ref.__gxx_personality_v0
75  ---------- 0x000050c0     GLOBAL OBJ    8        stdin@GLIBC_2.2.5
83  ---------- 0x0000522c     GLOBAL OBJ    4        fruitX
102 ---------- 0x000050b8     GLOBAL OBJ    0        __TMC_END__
104 ---------- 0x00005224     GLOBAL OBJ    4        x
117 ---------- 0x00005100     GLOBAL OBJ    272      std::cout
120 0x000050a8 0x5583cd5530a8 GLOBAL OBJ    4        width
124 ---------- 0x00005228     GLOBAL OBJ    4        y
126 ---------- 0x00005240     GLOBAL OBJ    24       tail
149 ---------- 0x00005220     GLOBAL OBJ    4        dir
164 ---------- 0x00005234     GLOBAL OBJ    4        score
```

À ce moment-là, le plus simple serait d'agir sur la variable score en modifiant sa valeur pendant l'exécution du code :

#GDB

Etape 1 :

Dans un terminal.

```
./snake
```

Etape 2:

Dans un second terminal, récupérer le pid d'exécution de snake.

```
pgrep snake
13500
```

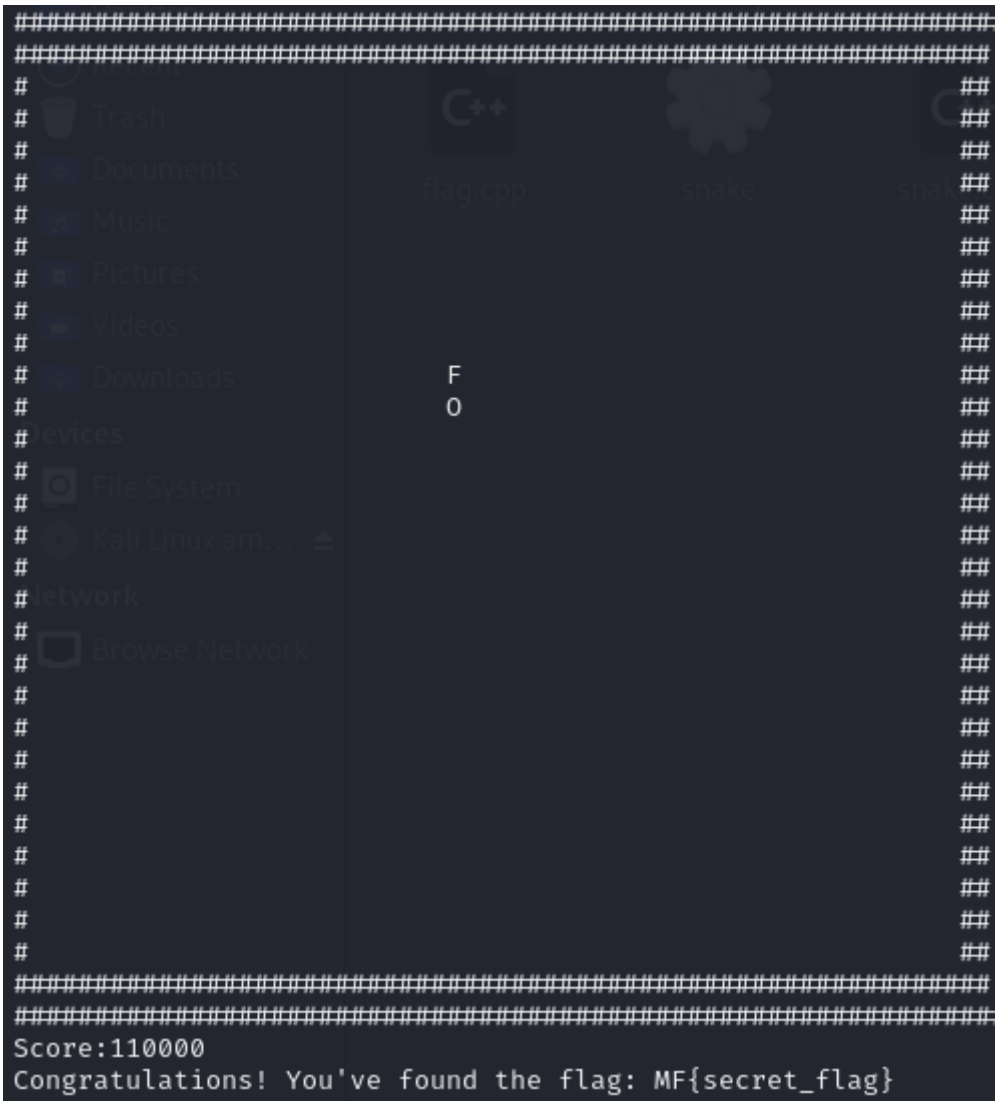
Etape 3:
Lancer gdb en le rattachant au processus snake.

```
sudo gdb -p 13500
```

Etape 4:
Modification de la variable score et continuer l'exécution du jeu.

```
(gdb) set variable score=110000
(gdb) continue
Continuing.
```

Etape 5:
Retourner sur le terminal exécutant snake, manger un fruit, le flag apparaît.


F
O
Score:110000
Congratulations! You've found the flag: MF{secret_flag}