gdb testflag

```
root@kali:~/Downloads/113-reversing-2# gdb testflag
GNU gdb (Debian 7.12-6) 7.12.0.20161007-git
Copyright (C) 2016 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from testflag...(no debugging symbols found)...done.
(gdb)
```

disas main

```
(gdb) disas main
Dump of assembler code for function main:
   0x080484cb <+0>:     lea    0x4(%esp),%ecx
   0x080484cf <+4>:     and    $0xfffffff0,%esp
   0x080484d2 <+7>:     pushl  -0x4(%ecx)
   0x080484d5 <+10>:    push   %ebp
   0x080484d6 <+11>:    mov    %esp,%ebp
   0x080484d8 <+13>:    push   %ebx
   0x080484d9 <+14>:    push   %ecx
   0x080484da <+15>:    sub    $0x40,%esp
   0x080484dd <+18>:    mov    %ecx,%ebx
   0x080484df <+20>:    mov    0x4(%ebx),%eax
   0x080484e2 <+23>:    mov    %eax,-0x3c(%ebp)
   0x080484e5 <+26>:    mov    %gs:0x14,%eax
   0x080484eb <+32>:    mov    %eax,-0xc(%ebp)
   0x080484ee <+35>:    xor    %eax,%eax
   0x080484f0 <+37>:    movl   $0x3,-0x38(%ebp)
   0x080484f7 <+44>:    push   $0x0
   0x080484f9 <+46>:    push   $0x1
   0x080484fb <+48>:    push   $0x0
   0x080484fd <+50>:    push   $0x0
   0x080484ff <+52>:    call   0x80483c0 <ptrace@plt>
   0x08048504 <+57>:    add    $0x10,%esp
   0x08048507 <+60>:    cmp    $0xffffffff,%eax
---Type <return> to continue, or q <return> to quit---
   0x0804850a <+63>:    jne    0x8048526 <main+91>
   0x0804850c <+65>:    sub    $0xc,%esp
   0x0804850f <+68>:    push   $0x8048680
   0x08048514 <+73>:    call   0x8048390 <puts@plt>
   0x08048519 <+78>:    add    $0x10,%esp
```

saut vers la détection du debugger a l'adresse : 0x0804850a (vu le jump avec iDA et control flow graph)

On met un breakpoint sur le cmp pour esquiver le saut vers le mouhaha

b* 0x08048507 (addresse cmp before jnz qui emmène au mouhahah : detecttion du debugger)
jump *0x8048526 (pour ne pas aller au mouhahah)
b *0x080485a3 (adresse du call strcmp, endroit de comparaison du flag et de notre input)
strcmp prend en arg eax et edx (eax notre input , edx le flag)

```
(gdb) b* 0x08048507
Breakpoint 1 at 0x8048507
(gdb) b *0x080485a3
Breakpoint 2 at 0x80485a3
(gdb) run aaaaa
Starting program: /root/.local/share/Trash/files/113-reversing-2/testflag aaaaa

Breakpoint 1, 0x08048507 in main ()
(gdb) jump *0x8048526
Continuing at 0x8048526.

Breakpoint 2, 0x080485a3 in main ()
(gdb)
```

valeur de edx :
info registers

```
(gdb) info registers
eax            0xffffd4cc       -11060
ecx            0x42     66
edx            0xffffd22b       -11733
ebx            0xffffd270       -11664
esp            0xffffd200       0xffffd200
ebp            0xffffd258       0xffffd258
esi            0xf7fa9000       -134574080
edi            0xf7fa9000       -134574080
eip            0x80485a3        0x80485a3 <main+216>
eflags         0x292    [ AF SF IF ]
cs             0x23     35
ss             0x2b     43
ds             0x2b     43
es             0x2b     43
fs             0x0      0
gs             0x63     99
(gdb)
```

adresse edx: 0xffffd22b

x/s 0xffffd22b: uLcTkBsJaRiZqHyPgXoFwNeVmDuLcTkB

```
(gdb)  x/s 0xffffd22b
0xffffd22b:     "uLcTkBsJaRiZqHyPgXoFwNeVmDuLcTkB"
(gdb)
```

```
root@kali:~/Downloads/113-reversing-2# ./testflag uLcTkBsJaRiZqHyPgXoFwNeVmDuLcTkB
Welcome!
```