FONAMENTS D'ORDINADORS

TEMA 8: Eines frikis per ASM

Manel Guerrero

Compilant per debugar

Per compilar per debugar em de posar el «-g» perquè inclogui els debugging symbols.

A part, anirà bé que posem l'optimització amb «-O» a '0' o a 'g'.

«-Og» està en el gcc a partir de la versió 4.8 i es «optimize for debugging»

```
$ gcc -g -O0 -c foo.c

$ gcc -g -Og -c foo.c

$ objdump -d -M suffix foo.o
```

C, assemblador i màquina

```
$ cat foo.c
int a=0;
main(){
a = a + 2;
}
$ gcc -g -c foo.c
$ objdump -d -M suffix foo.o

foo.o: file format elf32-i386



Disassembly of section .text:
```

```
00000000 <main>:
   0: 55                    pushl  %ebp
   1: 89 e5                 movl   %esp,
%ebp
   3: a1 00 00 00 00        movl
0x0,%eax
   8: 83 c0 02              addl
$0x2,%eax
   b: a3 00 00 00 00        movl
%eax,0x0
  10: 5d                    popl   %ebp
  11: c3                    retl
$
```

C, assemblador i màquina (2)

```
$ objdump -S -M suffix foo.o
foo.o:     file format elf32-i386
Disassembly of section .text:
00000000 <main>:
int a=0;
main(){
   0: 55                    pushl  %ebp
   1: 89 e5                 movl   %esp,%ebp
a = a + 2;
   3: a1 00 00 00 00        movl   0x0,%eax
   8: 83 c0 02              addl   $0x2,%eax
   b: a3 00 00 00 00        movl   %eax,0x0
}
  10: 5d                    popl   %ebp
  11: c3                    retl
```

C, assemblador i màquina (3)

```
$ gcc -g -o foo foo.c
$ gdb foo
(gdb) disassemble main
Dump of assembler code for function main:
   0x080483b4 <+0>: push   %ebp
   0x080483b5 <+1>: mov    %esp,%ebp
   0x080483b7 <+3>: mov    0x804a018,%eax
   0x080483bc <+8>: add    $0x2,%eax
   0x080483bf <+11>:mov    %eax,0x804a018
   0x080483c4 <+16>:pop    %ebp
   0x080483c5 <+17>:ret
End of assembler dump.
(gdb)
```

C, assemblador i màquina (4)

```
$ gcc -S foo.c
$ cat foo.s
        .file   "foo.c"
        .globl  a
        .bss
        .align 4
        .type   a, @object
        .size   a, 4
a:
        .zero   4
        .text
        .globl  main
        .type   main, @function
main:
.LFB0:
        .cfi_startproc
        pushl   %ebp
        .cfi_def_cfa_offset 8
        .cfi_offset 5, -8
        movl %esp, %ebp
        .cfi_def_cfa_register 5
        movl a, %eax
        addl $2, %eax
        movl %eax, a
        popl %ebp
        .cfi_def_cfa 4, 4
        .cfi_restore 5
        ret
        .cfi_endproc
.LFE0:
        .size  main, .-main
        .ident "GCC: (Ubuntu/Linaro 4.6.3-
1ubuntu5) 4.6.3"
        .section .note.GNU-stack,"",@progbits
$
```

GDB per veure registres i flags

```
$ cat flags.s
.bss
.comm a, 4, 1
.comm b, 4, 1
.text
.global main
main:
movb $1, a
movb $-1, b
movb a, %al
subb b, %al
movl $0, %ebx
movl $1, %eax
int $0x80
$
```

```
$ gcc -g -o flags flags.s
$ gdb flags
(gdb) break main
(gdb) run
(gdb) next [ret] 9 movb $-1, b
(gdb) next [ret] 10 movb a, %al
(gdb) next [ret] 11 subb b, %al
(gdb) next [ret] 13 movl $0, %ebx
(gdb) info registers
eax            0x2 2
[...]
eflags         0x213[ CF AF IF ]
[...]
(gdb) quit
```