Київський національний університет імені Тараса Шевченка факультет радіофізики, електроніки та комп'ютерних систем

Лабораторна робота №4

Роботу виконав
Студент 3-го курсу
ФРЕКС КІ-МА
Паньшин Дмитро

Київ 2020

1.

Підготовка середовища

```
sudo yum group install "Development Tools"
```

```
[root@g00-s00 ~]# gcc --version
gcc (GCC) 4.8.5 20150623 (Red Hat 4.8.5-39)
Copyright (C) 2015 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

[root@g00-s00 ~]#
[root@g00-s00 ~]# gdb --version
GNU gdb (GDB) Red Hat Enterprise Linux 7.6.1-119.el7
Copyright (C) 2013 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-redhat-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
[root@g00-s00 ~]#
[root@g00-s00 ~]# make --version
GNU Make 3.82
Built for x86_64-redhat-linux-gnu
Copyright (C) 2010  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.
[root@g00-s00 ~]# _
[root@g00-s00 ~]# as --version
GNU assembler version 2.27-41.base.el7_7.2
Copyright (C) 2016 Free Software Foundation, Inc.
This program is free software; you may redistribute it under the terms of
the GNU General Public License version 3 or later.
This program has absolutely no warranty.
This assembler was configured for a target of `x86_64-redhat-linux'.
[root@g00-s00 ~]# _
```

Завантажуємо файли за допомогою команди:wget

```
[root@g00-s00 lab4]# wget http://tilde.slu.kiev.ua/cs/asm/defs.h
--2020-06-03 09:57:07--  http://tilde.slu.kiev.ua/cs/asm/defs.h
Resolving tilde.slu.kiev.ua (tilde.slu.kiev.ua)... 91.202.129.232
Connecting to tilde.slu.kiev.ua (tilde.slu.kiev.ua)|91.202.129.232|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 887 [application/octet-stream]
Saving to: ‘defs.h’

100%[======================================================>] 887         --.-K/s   in 0s

2020-06-03 09:57:07 (51.1 MB/s) - ‘defs.h’ saved [887/887]

[root@g00-s00 lab4]# ls
defs.h
[root@g00-s00 lab4]# wget http://tilde.slu.kiev.ua/cs/asm/exit.s
--2020-06-03 09:57:32--  http://tilde.slu.kiev.ua/cs/asm/exit.s
Resolving tilde.slu.kiev.ua (tilde.slu.kiev.ua)... 91.202.129.232
Connecting to tilde.slu.kiev.ua (tilde.slu.kiev.ua)|91.202.129.232|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 105 [application/octet-stream]
Saving to: ‘exit.s’

100%[======================================================>] 105         --.-K/s   in 0s

2020-06-03 09:57:32 (15.6 MB/s) - ‘exit.s’ saved [105/105]

[root@g00-s00 lab4]# ls
defs.h  exit.s
[root@g00-s00 lab4]#
```

Виконайте асемблювання програми-заготовки та зв'язування:

```
[root@g00-s00 lab4]# as -o exit.o -c exit.s

[root@g00-s00 lab4]# ld -static -o exit exit.o
[root@g00-s00 lab4]# _
```

2. Автоматизація збірки

Створіть Makefile, який за командою `make exit` та `make all` виконає збірку, а за командою `make clean` очистить об'єктні та виконувані файли.
Модифікуйте Makefile так, щоб опції асемблера та лінкера задавалися змінними `ASFLAGS` та `LDFLAGS`.
Додайте опцію асемблера для генерації відлагоджувальних символів DWARF.
Використайте шаблонні правила так, щоб можна було збирати декілька асемблерних файлів в окремі виконувані файли. Це знадобиться при виконанні індивідуального завдання.

```
AS_FLAGS=--gdwarf-2 -o
LD_FLAGS=--static
SOURC=task4.s
OBJ=$(SOURC:.s=.o)
all: $(SOURC) $(EXECUTE)
$(EXECUTE): $(OBJ)
   ld $(LD_FLAGS) $(OBJ) -o $@
.s.o:
   as $(AS_FLAGS) $@ -c $<
clean:
   rm -f *.o
```

3. Навички відлагоджування

Завантажте одержаний виконуваний файл у відлагоджувач за допомогою команди:
`gdb ./exit`
Встановіть точку зупинки на початок програми (мітка _start):
`b _start`
Запустіть програму
`run`
Після зупинки виконання програми перегляньте вміст регістрів:
`info registers` або `i r`
Переходьте до виконання наступної команди:
`next` або `n`
Для виходу із режиму покрокового виконання використовуйте команду
`continue` або `c`
Програма працюватиме до наступної точки зупинки або до повного чи аварійного завершення.
Для перегляду адресного простору процесу скористайтесь командою x, наприклад:
`x/16gx 0x12345678`

```
[root@g00-s00 lab4]# gdb ./exit
GNU gdb (GDB) Red Hat Enterprise Linux 7.6.1-119.el7
Copyright (C) 2013 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-redhat-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from /root/lab4/exit...(no debugging symbols found)...done.
(gdb)
```

```
(gdb) b _start
Breakpoint 1 at 0x400078
(gdb) run
Starting program: /root/lab4/./exit

Breakpoint 1, 0x0000000000400078 in _start ()
(gdb) _
```

```
(gdb) info registers
rax            0x0      0
rbx            0x0      0
rcx            0x0      0
rdx            0x0      0
rsi            0x0      0
rdi            0x0      0
rbp            0x0      0x0
rsp            0x7fffffffe670   0x7fffffffe670
r8             0x0      0
r9             0x0      0
r10            0x0      0
r11            0x0      0
r12            0x0      0
r13            0x0      0
r14            0x0      0
r15            0x0      0
rip            0x400078 0x400078 <_start>
eflags         0x202    [ IF ]
cs             0x33     51
ss             0x2b     43
ds             0x0      0
es             0x0      0
fs             0x0      0
gs             0x0      0
(gdb)
```

```
(gdb) next
Single stepping until exit from function _start,
which has no line number information.
[Inferior 1 (process 20658) exited normally]
(gdb) continue
```

4. Контрольований fork-вибух

Створіть програму, яка "розмножується" за допомогою fork() 5 разів.
Кожна нова копія виводить свою генерацію (число від 1 до 5) на стандартний вивід та завершується.
Батьківський процес очікує завершення дочірнього та також завершується.

Псевдокод

```
byte nforks = 5;
main() {
    quad pid;
    byte n;
fork:
    pid = fork();
    if(pid != 0) goto parent;
child:
    n = nforks + 0x30;      /* 1 + 0x30 = '1' */
    write(stdout, 1, &n);

    nforks--;
    if(nforks == 0) goto end;
    goto fork;
parent:
    wait4(pid, 0, 0, 0);
end:
    exit(0);
}
```

Перевірка

В результаті ви маєте отримати вивід: `54321`

Код программи:

```
.include "defs.h"_

.section .bss
n: .byte 0

.section .data
newline: .byte '\n'
.global _start

_start:

        movq $5, %r9    /*nfork==5*/

fork:
        movq $SYS_FORK, %rax
        syscall
        movq %rax, %r13

        cmpq $0, %r13   /*if pid==0 go to parent*/
        jne parent

child:
        leaq 0x30(%r9), %r11    /*1+0x30 = '1'*/
        movq %r11, n

        /*Writing numbers*/
        movq $SYS_WRITE, %rax
        movq $STDOUT, %rdi
        movq $n, %rsi
        movq $1, %rdx
        syscall

        decq %r9

        cmpq $0, %r9    /*if(nfoks ==0) go to end*/
        je end
-- INSERT --


        cmpq $0, %r9    /*if(nfoks ==0) go to end*/
        je end

        jmp fork

parent:
        movq $SYS_WAIT4, %rax
        movq %r13, %rdi
        movq $0, %rsi
        movq $0, %rdx
        movq $0, %r13
        syscall
end:
        movq $SYS_WRITE, %rax
        movq $STDOUT, %rdi
        movq $1, %rdx
        movq $newline, %rsi
        syscall                 /*write on newline*/
        movq $SYS_EXIT, %rax
        xorq %rdi, %rdi
        syscall         /*exit*/_
-- INSERT --                                                    55,
```

Робота make file та самої программи:

```
[root@g00-s00 lab4]# ls
defs.h  exit  exit.s  makefile  task4  task4.s
[root@g00-s00 lab4]# make all
as --gdwarf-2 -o task4.o -c task4.s
ld --static task4.o -o task4
[root@g00-s00 lab4]# ./task4
54321




[root@g00-s00 lab4]# make clean
rm -f *.o
[root@g00-s00 lab4]# ls
defs.h  exit  exit.s  makefile  task4  task4.s
[root@g00-s00 lab4]# _
```

Debugging with gdb:

```
[root@g00-s00 lab4]# gdb ./task4
GNU gdb (GDB) Red Hat Enterprise Linux 7.6.1-119.el7
Copyright (C) 2013 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-redhat-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from /root/lab4/task4...done.
(gdb) b 18
Breakpoint 1 at 0x60008c: file task4.s, line 18.
(gdb) r
Starting program: /root/lab4/./task4
[Detaching after fork from child process 4052]

Breakpoint 1, 0x000000000060008c in fork ()
(gdb) 54321
```

```
i r
rax            0xfd4    4052
rbx            0x0      0
rcx            0xffffffffffffffff       -1
rdx            0x0      0
rsi            0x0      0
rdi            0x0      0
rbp            0x0      0x0
rsp            0x7fffffffe670   0x7fffffffe670
r8             0x0      0
r9             0x5      5
r10            0x0      0
r11            0x202    514
r12            0x0      0
r13            0xfd4    4052
r14            0x0      0
r15            0x0      0
rip            0x60008c 0x60008c
eflags         0x202    [ IF ]
cs             0x33     51
ss             0x2b     43
ds             0x0      0
es             0x0      0
fs             0x0      0
gs             0x0      0
```

Висновки: під час виконання лабораторної роботи були здобуті навички з основ Assembler для архітектури x86_64, також було розроблено програму для створення системного виклику fork процесів, відлагоджено за допомогою утиліти для налагодження GDB. Був створений Makefile для автоматизації компіляції та зв’язування програми.

Репозитрорій: https://github.com/N0thingLikeTheSun/CS_LABS