Лабораторная работа №15. Использование системных вызовов Linux x86_64 и стандартной библиотеки Си. Статическая и динамическая компоновка

Макаров П. А.

1 Краткая теория

1. Ray Toal NASM Tutorial.
2. Linking C with NASM.
3. NASM manual.
4. Intel 64 and IA-32 Architectures Software Developer Manuals.
5. Marcin Juszkiewicz Linux kernel system calls for all architectures.
6. Filippo Valsorda Searchable Linux Syscall Table for x86 and x86_64.
7. Linus Torvalds linux/arch/x86/entry/syscalls/syscall_64.tbl.

2 Задания для самостоятельного решения

Напишите в текстовом редакторе `vim` следующие исходные тексты программ. Ассемблируйте их, выполните статическую и динамическую компоновку. Запустите получившиеся программы и исследуйте их с помощью утилит `file`, `time`, `hexdump`, `objdump`, `ldd` и `readelf`. Сравните результаты между собой, и с тем, что получалось в предыдущей лабораторной работе для файла `hello.c`.

1. Пример с использованием системных вызовов.

Листинг 1: Содержимое файла `hello.asm`

```
; ------------------------------------------------------------
; Write "Hello, world!" to the stdout using system calls.
; To assemble:
;   nasm -f elf64 hello.asm
; To link:
;   ld hello.o -o hello
; ------------------------------------------------------------

global  _start

section .text
_start: mov     rax, 1                  ; system call for write
        mov     rdi, 1                  ; handle 1 is stdout
        mov     rsi, msg                ; address of string
        mov     rdx, len                ; number of bytes
        syscall                         ; invoke OS to write
        mov     rax, 60                 ; system call for exit
        xor     rdi, rdi                ; exit code 0
        syscall                         ; invoke OS to exit

section .data
msg:    db      "Hello, world!", 0x0A   ; string + newline char
len     equ     $-msg
```

2. Пример с использованием стандартной библиотеки Си (статическая компоновка).

Листинг 2: Содержимое файла `puts1.asm`

```
; ------------------------------------------------------------
; Write "Hello, world!" to the stdout using C library.
; Static linking
; To assemble:
;   nasm -f elf64 puts1.asm
; To link:
;   gcc puts1.o -o puts1 -static
; ------------------------------------------------------------

extern  puts

global  main

section .text
main:   mov  rdi, msg
        call puts
        ret

section .data
msg:    db   "Hello, world!", 0
```

3. Пример с использованием стандартной библиотеки Си (динамическая компоновка).

Листинг 3: Содержимое файла `puts2.asm`

```
; ------------------------------------------------------------
; Write "Hello, world!" to the stdout using C library.
; Dynamic linking
; To assemble:
;   nasm -f elf64 puts2.asm
; To link:
;   ld puts2.o -lc -I /lib64/ld-linux-x86-64.so.2 -o puts2
; ------------------------------------------------------------

extern  puts
extern  exit

global  _start

section .text
_start: mov  rdi, msg
        call puts

        push 0
        call exit

section .data
msg:    db   "Hello, world!", 0
```