Московский авиационный институт (национальный исследовательский университет)

# Факультет информационных технологий и прикладной математики

**Кафедра вычислительной математики и программирования**

**Лабораторная работа №1 по курсу «Операционные системы»**

**Приобретение практических навыков диагностики работы программного обеспечения.**

Студент: Селивёрстов Д. С. Преподаватель: Миронов Е. С.

Группа: М8О-201Б-21

Дата: Оценка: Подпись:

**Москва, 2023**

# Условие

Ознакомиться с сигналами операционной системы UNIX/LINUX, используя утилиту strace, проанализировать результаты, сопоставить их с кодом программы.

# Метод решения

Использовать свободно распространяемую утилиту strace следующим образом: strace lab2

# Вывод strace

execve("./lab2/child", ["./lab2/child"], 0x7ffcb1915590 /\* 60 vars \*/) = 0

brk(NULL) = 0x555ca9e00000

arch\_prctl(0x3001 /\* ARCH\_??? \*/, 0x7ffd51213220) = -1 EINVAL (Invalid argument)

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f0c41c57000

access("/etc/ld.so.preload", R\_OK) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/etc/ld.so.cache", O\_RDONLY|O\_CLOEXEC) = 3

newfstatat(3, "", {st\_mode=S\_IFREG|0644, st\_size=73031, ...}, AT\_EMPTY\_PATH) = 0

mmap(NULL, 73031, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7f0c41c45000

close(3) = 0

openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/libc.so.6", O\_RDONLY|O\_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0P\237\2\0\0\0\0\0"..., 832) = 832

pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784

pread64(3, "\4\0\0\0 \0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0"..., 48, 848) = 48

pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0i8\235HZ\227\223\333\350s\360\352,\223\340."..., 68, 896) = 68

newfstatat(3, "", {st\_mode=S\_IFREG|0644, st\_size=2216304, ...}, AT\_EMPTY\_PATH) = 0

pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784

mmap(NULL, 2260560, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f0c41a1d000

mmap(0x7f0c41a45000, 1658880, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x28000) = 0x7f0c41a45000

mmap(0x7f0c41bda000, 360448, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1bd000) = 0x7f0c41bda000

mmap(0x7f0c41c32000, 24576, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x214000) = 0x7f0c41c32000

mmap(0x7f0c41c38000, 52816, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7f0c41c38000

close(3) = 0

mmap(NULL, 12288, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f0c41a1a000

arch\_prctl(ARCH\_SET\_FS, 0x7f0c41a1a740) = 0

set\_tid\_address(0x7f0c41a1aa10) = 8579

set\_robust\_list(0x7f0c41a1aa20, 24) = 0

rseq(0x7f0c41a1b0e0, 0x20, 0, 0x53053053) = 0

mprotect(0x7f0c41c32000, 16384, PROT\_READ) = 0

mprotect(0x555ca9167000, 4096, PROT\_READ) = 0

mprotect(0x7f0c41c91000, 8192, PROT\_READ) = 0

prlimit64(0, RLIMIT\_STACK, NULL, {rlim\_cur=8192\*1024, rlim\_max=RLIM64\_INFINITY}) = 0

munmap(0x7f0c41c45000, 73031) = 0

getrandom("\xd3\xc0\xf8\x18\x2a\x45\x93\x3e", 8, GRND\_NONBLOCK) = 8

brk(NULL) = 0x555ca9e00000

brk(0x555ca9e21000) = 0x555ca9e21000

newfstatat(0, "", {st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0x2), ...}, AT\_EMPTY\_PATH) = 0

read(0, somestring "somestring\n", 1024)

"somestring \"somestring\\n\", 1024)"..., 1024) = 33

write(1, "smstrng \"smstrng\\n\", 1024)\n", 27smstrng "smstrng\n", 1024)

) = 27

read(0, shortstring545 "shortstring545\n", 1024)

"shortstring545 \"shortstring545\\n"..., 1024) = 41

write(1, "shrtstrng545 \"shrtstrng545\\n\", 1"..., 37shrtstrng545 "shrtstrng545\n", 1024)

) = 37

read(0, "", 1024)

"\"\", 1024)\n", 1024) = 10

write(1, "\"\", 1024)\n", 10"", 1024)

) = 10

read(0, "", 1024) = 0

write(1, "", 0) = 0

exit\_group(0) = ?

+++ exited with 0 +++

# Выводы

Вызов *fork* дублирует породивший его процесс со всеми его переменными, файловыми дескрипторами, приоритетами процесса, рабочий и корневой каталоги, и сегментами выделенной памяти.

Ребёнок **не** наследует:

* идентификатора процесса (PID, PPID);
* израсходованного времени ЦП (оно обнуляется);
* сигналов процесса-родителя, требующих ответа;
* блокированных файлов (record locking).

В процессе выполнения лабораторной работы были приобретены навыки практи- ческого применения создания, обработки и отслеживания их состояния. Для выполне- ния данного варианта задания создание потоков как таковых не требуется, так как всю работу выполняет системный вызов «exec».