Московский Авиационный Институт

(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

**Лабораторная работа № 1 по курсу**

**«Операционные системы»**

Студент: Постнов Александр Вячеславович

Группа: М8О-201Б-21

Преподаватель: Миронов Евгений Сергеевич

Оценка: \_\_\_\_\_\_\_\_\_\_\_

Дата: \_\_\_\_\_\_\_\_\_\_\_

Подпись: \_\_\_\_\_\_\_\_\_\_\_

Москва, 2022

**Содержание**

1. Репозиторий
2. Постановка задачи
3. Общий метод и алгоритм решения
4. Выводы

**Репозиторий**

https://github.com/kappaprideonly/mai-os-labs

**Постановка задачи**

Продемонстрировать работу утилиты *strace* на примере лабораторной работы

**Общие сведения о программе**

*strace* — это утилита, отслеживающая системные вызовы, которые представляют собой механизм трансляции, обеспечивающий интерфейс между процессом и операционной системой (ядром). Эти вызовы могут быть перехвачены и прочитаны. Это позволяет лучше понять, что процесс пытается сделать в заданное время. Перехватывая эти вызовы, мы можем добиться лучшего понимания поведения процессов, особенно если что-то идет не так.

**Общий метод и алгоритм решения**Я продемонстрирую работу утилиты strace на примере 2 лабораторной работы.

В этой работе я использовал следующие системные вызовы: pipe, fork, dup2, execl

Strace показал все системные вызовы, а так вызовы чтения и записи (вызовы read/write число прочитанных/записанных байт). fork вызывает clone для создания дочернего процесса.

➜ lab2 git:(main) strace ./lab2 <test.txt

execve("./lab2", ["./lab2"], 0x7ffed75067e0 /\* 64 vars \*/) = 0

brk(NULL) = 0x5633c2c09000

arch\_prctl(0x3001 /\* ARCH\_??? \*/, 0x7ffc483ce090) = -1 EINVAL (Недопустимый аргумент)

access("/etc/ld.so.preload", R\_OK) = -1 ENOENT (Нет такого файла или каталога)

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

newfstatat(3, "", {st\_mode=S\_IFREG|0644, st\_size=148091, ...}, AT\_EMPTY\_PATH) = 0

mmap(NULL, 148091, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7f72ae04a000

close(3) = 0

openat(AT\_FDCWD, "/usr/lib/libstdc++.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\0\0\0\0\0\0\0\0\0"..., 832) = 832

newfstatat(3, "", {st\_mode=S\_IFREG|0755, st\_size=19198496, ...}, AT\_EMPTY\_PATH) = 0

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f72ae048000

mmap(NULL, 2320384, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f72ade11000

mmap(0x7f72adeaa000, 1138688, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x99000) = 0x7f72adeaa000

mmap(0x7f72adfc0000, 487424, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1af000) = 0x7f72adfc0000

mmap(0x7f72ae037000, 57344, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x225000) = 0x7f72ae037000

mmap(0x7f72ae045000, 10240, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7f72ae045000

close(3) = 0

openat(AT\_FDCWD, "/usr/lib/libm.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\0\0\0\0\0\0\0\0\0"..., 832) = 832

newfstatat(3, "", {st\_mode=S\_IFREG|0755, st\_size=944600, ...}, AT\_EMPTY\_PATH) = 0

mmap(NULL, 946368, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f72add29000

mmap(0x7f72add37000, 499712, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0xe000) = 0x7f72add37000

mmap(0x7f72addb1000, 385024, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x88000) = 0x7f72addb1000

mmap(0x7f72ade0f000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0xe5000) = 0x7f72ade0f000

close(3) = 0

openat(AT\_FDCWD, "/usr/lib/libgcc\_s.so.1", O\_RDONLY|O\_CLOEXEC) = 3

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

newfstatat(3, "", {st\_mode=S\_IFREG|0644, st\_size=571848, ...}, AT\_EMPTY\_PATH) = 0

mmap(NULL, 127304, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f72add09000

mmap(0x7f72add0c000, 94208, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x3000) = 0x7f72add0c000

mmap(0x7f72add23000, 16384, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1a000) = 0x7f72add23000

mmap(0x7f72add27000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1d000) = 0x7f72add27000

close(3) = 0

openat(AT\_FDCWD, "/usr/lib/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\0P4\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

newfstatat(3, "", {st\_mode=S\_IFREG|0755, st\_size=1953472, ...}, 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, 1994384, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f72adb22000

mmap(0x7f72adb44000, 1421312, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x22000) = 0x7f72adb44000

mmap(0x7f72adc9f000, 356352, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x17d000) = 0x7f72adc9f000

mmap(0x7f72adcf6000, 24576, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1d4000) = 0x7f72adcf6000

mmap(0x7f72adcfc000, 52880, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7f72adcfc000

close(3) = 0

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f72adb20000

arch\_prctl(ARCH\_SET\_FS, 0x7f72adb21200) = 0

set\_tid\_address(0x7f72adb214d0) = 37608

set\_robust\_list(0x7f72adb214e0, 24) = 0

rseq(0x7f72adb21b20, 0x20, 0, 0x53053053) = 0

mprotect(0x7f72adcf6000, 16384, PROT\_READ) = 0

mprotect(0x7f72add27000, 4096, PROT\_READ) = 0

mprotect(0x7f72ade0f000, 4096, PROT\_READ) = 0

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f72adb1e000

mprotect(0x7f72ae037000, 53248, PROT\_READ) = 0

mprotect(0x5633c2424000, 4096, PROT\_READ) = 0

mprotect(0x7f72ae0a0000, 8192, PROT\_READ) = 0

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

munmap(0x7f72ae04a000, 148091) = 0

getrandom("\x6c\x75\xfa\x2f\x8c\xf7\xcd\x3e", 8, GRND\_NONBLOCK) = 8

brk(NULL) = 0x5633c2c09000

brk(0x5633c2c2a000) = 0x5633c2c2a000

futex(0x7f72ae0456bc, FUTEX\_WAKE\_PRIVATE, 2147483647) = 0

newfstatat(0, "", {st\_mode=S\_IFREG|0644, st\_size=31, ...}, AT\_EMPTY\_PATH) = 0

read(0, "checker.txt\n1 2 3 4 56\n100 2 5\n", 4096) = 31

pipe2([3, 4], 0) = 0

clone(child\_stack=NULL, flags=CLONE\_CHILD\_CLEARTID|CLONE\_CHILD\_SETTID|SIGCHLD, child\_tidptr=0x7f72adb214d0) = 37609

close(3) = 0

write(4, "1 2 3 4 56\n", 11) = 11

write(4, "100 2 5\n", 8) = 8

read(0, "", 4096) = 0

close(4) = 0

wait4(-1, NULL, 0, NULL) = 37609

--- SIGCHLD {si\_signo=SIGCHLD, si\_code=CLD\_EXITED, si\_pid=37609, si\_uid=1000, si\_status=0, si\_utime=0, si\_stime=0} ---

exit\_group(0) = ?

+++ exited with 0 ++

**Выводы:**

Во время выполнения работы я изучил утилиту strace, понял, что она может быть полезной во время поиска ошибок.