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: Марченко Алексей Эдуардович

Группа: М8О-207Б-21

Преподаватель: Миронов Евгений Сергеевич

Оценка: \_\_\_\_\_\_\_\_\_\_\_

Дата: \_\_\_\_\_\_\_\_\_\_\_

Подпись: \_\_\_\_\_\_\_\_\_\_\_

Москва, 2023

**Содержание**

1. Репозиторий
2. Постановка задачи

2.1 Цель работы

2.2 Задание

1. Системные вызовы
2. Демонстрация работы strace
3. Выводы

**Репозиторий**

https://github.com/MarchAleksey/OS\_Labs

**Постановка задачи**

**Цель работы**

Приобретение практических навыков диагностики работы программного обеспечения.

**Задание**

При выполнении последующих лабораторных работ необходимо продемонстрировать ключевые системные вызовы, которые в них используются и то, что их использование соответствует варианту ЛР.

**Системные вызовы**

**execve**

execve - это системный вызов в операционных системах Linux, который используется для выполнения новой программы в текущем процессе. Он заменяет текущий процесс новым процессом, загружая в него новую программу.

**mmap**

mmap (от англ. memory map) - это системный вызов в операционной системе Linux, используемый для отображения файла или устройства в память процесса.

**access**

access - это системный вызов в операционной системе Linux, который проверяет, существует ли файл с указанным путем и есть ли у текущего процесса права на его чтение/запись/исполнение. Аргументы, переданные в access: путь к файлу и режим доступа (READ, WRITE, EXECUTE) для проверки.

**newfstatat**

newfstatat - это системный вызов в операционной системе Linux, который позволяет получить информацию о файле или каталоге, на который ссылается относительный путь от заданного каталога.

**openat**

openat - это системный вызов в операционной системе Linux, который открывает файл или директорию по указанному пути относительно заданного каталога файловой системы, который определяется его файловым дескриптором. аргументы, переданные в openat: файловый дескриптор каталога, путь к файлу или директории, режим открытия файла (чтение, запись, создание), а также дополнительные флаги и права доступа.

**pread64**

pread64 - это системный вызов в операционной системе Linux, который используется для чтения данных из файла в указанный буфер без изменения текущей позиции указателя файла.

**mprotect**

mprotect - это системный вызов в операционной системе Linux, который используется для изменения атрибутов защиты памяти. Этот вызов используется, например, для обеспечения безопасности при работе с указателями на память, а также для реализации механизма Copy-on-Write (COW).

**getrandom**

getrandom - это системный вызов в операционной системе Linux, который используется для получения случайных из источника случайности ядра операционной системы. аргументы, переданные в getrandom: указатель на буфер для записи случайных данных, количество запрашиваемых байтов и флаги, указывающие, какие дополнительные параметры использовать для генерации случайных данных. Возвращаемое значение getrandom указывает на количество собранных случайных байтов.

**Демонстрация работы strace (ЛР 2)**

execve("./main", ["./main"], 0x7ffed473f3c0 /\* 57 vars \*/) = 0

brk(NULL) = 0x55605e96c000

arch\_prctl(0x3001 /\* ARCH\_??? \*/, 0x7ffd5ecb2880) = -1 EINVAL (Недопустимый аргумент)

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f4b30ef8000

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=67231, ...}, AT\_EMPTY\_PATH) = 0

mmap(NULL, 67231, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7f4b30ee7000

close(3) = 0

openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/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|0644, st\_size=2252096, ...}, AT\_EMPTY\_PATH) = 0

mmap(NULL, 2267328, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f4b30c00000

mmap(0x7f4b30c9a000, 1114112, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x9a000) = 0x7f4b30c9a000

mmap(0x7f4b30daa000, 454656, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1aa000) = 0x7f4b30daa000

mmap(0x7f4b30e19000, 57344, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x218000) = 0x7f4b30e19000

mmap(0x7f4b30e27000, 10432, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7f4b30e27000

close(3) = 0

openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/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=125488, ...}, AT\_EMPTY\_PATH) = 0

mmap(NULL, 127720, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f4b30ec7000

mmap(0x7f4b30eca000, 94208, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x3000) = 0x7f4b30eca000

mmap(0x7f4b30ee1000, 16384, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1a000) = 0x7f4b30ee1000

mmap(0x7f4b30ee5000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1d000) = 0x7f4b30ee5000

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) = 0x7f4b30800000

mmap(0x7f4b30828000, 1658880, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x28000) = 0x7f4b30828000

mmap(0x7f4b309bd000, 360448, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1bd000) = 0x7f4b309bd000

mmap(0x7f4b30a15000, 24576, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x214000) = 0x7f4b30a15000

mmap(0x7f4b30a1b000, 52816, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7f4b30a1b000

close(3) = 0

openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/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|0644, st\_size=940560, ...}, AT\_EMPTY\_PATH) = 0

mmap(NULL, 942344, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f4b30b19000

mmap(0x7f4b30b27000, 507904, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0xe000) = 0x7f4b30b27000

mmap(0x7f4b30ba3000, 372736, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x8a000) = 0x7f4b30ba3000

mmap(0x7f4b30bfe000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0xe4000) = 0x7f4b30bfe000

close(3) = 0

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f4b30ec5000

arch\_prctl(ARCH\_SET\_FS, 0x7f4b30ec63c0) = 0

set\_tid\_address(0x7f4b30ec6690) = 8625

set\_robust\_list(0x7f4b30ec66a0, 24) = 0

rseq(0x7f4b30ec6d60, 0x20, 0, 0x53053053) = 0

mprotect(0x7f4b30a15000, 16384, PROT\_READ) = 0

mprotect(0x7f4b30bfe000, 4096, PROT\_READ) = 0

mprotect(0x7f4b30ee5000, 4096, PROT\_READ) = 0

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f4b30ec3000

mprotect(0x7f4b30e19000, 45056, PROT\_READ) = 0

mprotect(0x55605cda7000, 4096, PROT\_READ) = 0

mprotect(0x7f4b30f32000, 8192, PROT\_READ) = 0

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

munmap(0x7f4b30ee7000, 67231) = 0

getrandom("\x83\xee\xea\x1b\x26\x65\xce\x03", 8, GRND\_NONBLOCK) = 8

brk(NULL) = 0x55605e96c000

brk(0x55605e98d000) = 0x55605e98d000

futex(0x7f4b30e2777c, FUTEX\_WAKE\_PRIVATE, 2147483647) = 0

newfstatat(0, "", {st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0x1), ...}, AT\_EMPTY\_PATH) = 0

read(0, "test.txt\n", 1024) = 9

openat(AT\_FDCWD, "test.txt", O\_RDONLY) = 3

pipe2([4, 5], 0) = 0

clone(child\_stack=NULL, flags=CLONE\_CHILD\_CLEARTID|CLONE\_CHILD\_SETTID|SIGCHLD, child\_tidptr=0x7f4b30ec6690) = 8799

read(4, "\n", 1) = 1

--- SIGCHLD {si\_signo=SIGCHLD, si\_code=CLD\_EXITED, si\_pid=8799, si\_uid=1000, si\_status=0, si\_utime=0, si\_stime=0} ---

read(4, "\232\231\331A", 4) = 4

newfstatat(1, "", {st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0x1), ...}, AT\_EMPTY\_PATH) = 0

write(1, "27.2\n", 5) = 5

read(4, "\377", 1) = 1

close(5) = 0

close(4) = 0

close(3) = 0

exit\_group(0) = ?

+++ exited with 0 +++

**Вывод**

Strace – это утилита Linux, отслеживающая системные вызовы, которые представляют собой механизм трансляции, обеспечивающий интерфейс между процессором и операционной системой.

Благодаря данной лабораторной работе, я приобрел навыки, необходимые для работы с strace, а также изучил системные вызовы.