МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ

ФЕДЕРАЦИИ

Московский Авиационный Институт

(Национальный Исследовательский Университет)

Институт №8 "Компьютерные науки и прикладная математика"

Кафедра 806 "Вычислительная математика и программирование"

Лабораторные работы №8

По курсу «Операционные системы»

Студент: Попов А. Д.

Группа: М8О-208Б-23

Преподаватель: Живалев Е. А.

Дата: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Москва, 2024

**Описание и определение strace:**

**strace —** это эффективный инструмент для диагностики и отладки, используемый в Unix-подобных операционных системах, таких как Linux. Он позволяет отслеживать и фиксировать системные вызовы, выполняемые процессом, а также сигналы, которые он обрабатывает. Системные вызовы представляют собой интерфейс взаимодействия между пользовательскими приложениями и ядром ОС. С помощью strace можно детально изучить, какие файлы открывает программа, какие сетевые соединения использует, какие ошибки возникают в процессе выполнения и многое другое. Этот инструмент особенно полезен для отладки, анализа производительности и изучения поведения процессов в реальном времени.

**Репозиторий:** https://github.com/aldpopov/OS\_labs/tree/master/LW8

**Листинг работы strace:**

*execve("./build/LW3/lab3", ["./build/LW3/lab3"], 0x7ffffa5e5c60 /\* 48 vars \*/) = 0*

*brk(NULL) = 0x5dfa34fbf000*

*mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x729412a4b000*

*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*

*fstat(3, {st\_mode=S\_IFREG|0644, st\_size=59691, ...}) = 0*

*mmap(NULL, 59691, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x729412a3c000*

*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\0\220\243\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*

*fstat(3, {st\_mode=S\_IFREG|0755, st\_size=2125328, ...}) = 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, 2170256, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x729412800000*

*mmap(0x729412828000, 1605632, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x28000) = 0x729412828000*

*mmap(0x7294129b0000, 323584, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1b0000) = 0x7294129b0000*

*mmap(0x7294129ff000, 24576, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1fe000) = 0x7294129ff000*

*mmap(0x729412a05000, 52624, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x729412a05000*

*close(3) = 0*

*mmap(NULL, 12288, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x729412a39000*

*arch\_prctl(ARCH\_SET\_FS, 0x729412a39740) = 0*

*set\_tid\_address(0x729412a39a10) = 6478*

*set\_robust\_list(0x729412a39a20, 24) = 0*

*rseq(0x729412a3a060, 0x20, 0, 0x53053053) = 0*

*mprotect(0x7294129ff000, 16384, PROT\_READ) = 0*

*mprotect(0x5dfa34281000, 4096, PROT\_READ) = 0*

*mprotect(0x729412a83000, 8192, PROT\_READ) = 0*

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

*munmap(0x729412a3c000, 59691) = 0*

*getrandom("\x39\x56\xf4\x4c\x41\x85\xee\x97", 8, GRND\_NONBLOCK) = 8*

*brk(NULL) = 0x5dfa34fbf000*

*brk(0x5dfa34fe0000) = 0x5dfa34fe0000*

*fstat(0, {st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0), ...}) = 0*

*read(0, "a\n", 1024) = 2*

*unlink("/dev/shm/sem.semaphore\_write") = 0*

*unlink("/dev/shm/sem.semaphore\_read") = 0*

*openat(AT\_FDCWD, "/dev/shm/sem.semaphore\_write", O\_RDWR|O\_NOFOLLOW|O\_CLOEXEC) = -1 ENOENT (No such file or directory)*

*getrandom("\x3a\x59\xc4\xd3\xcb\xb4\x1d\xc7", 8, GRND\_NONBLOCK) = 8*

*newfstatat(AT\_FDCWD, "/dev/shm/sem.wWTIG7", 0x7fff4f4c1640, AT\_SYMLINK\_NOFOLLOW) = -1 ENOENT (No such file or directory)*

*openat(AT\_FDCWD, "/dev/shm/sem.wWTIG7", O\_RDWR|O\_CREAT|O\_EXCL|O\_NOFOLLOW|O\_CLOEXEC, 0666) = 3*

*write(3, "\0\0\0\0\0\0\0\0\200\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0", 32) = 32*

*mmap(NULL, 32, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) = 0x729412a4a000*

*link("/dev/shm/sem.wWTIG7", "/dev/shm/sem.semaphore\_write") = 0*

*fstat(3, {st\_mode=S\_IFREG|0664, st\_size=32, ...}) = 0*

*unlink("/dev/shm/sem.wWTIG7") = 0*

*close(3) = 0*

*openat(AT\_FDCWD, "/dev/shm/sem.semaphore\_read", O\_RDWR|O\_NOFOLLOW|O\_CLOEXEC) = -1 ENOENT (No such file or directory)*

*getrandom("\xf9\xa8\x98\x25\x1c\x68\x8f\x59", 8, GRND\_NONBLOCK) = 8*

*newfstatat(AT\_FDCWD, "/dev/shm/sem.7rScGO", 0x7fff4f4c1640, AT\_SYMLINK\_NOFOLLOW) = -1 ENOENT (No such file or directory)*

*openat(AT\_FDCWD, "/dev/shm/sem.7rScGO", O\_RDWR|O\_CREAT|O\_EXCL|O\_NOFOLLOW|O\_CLOEXEC, 0666) = 3*

*write(3, "\0\0\0\0\0\0\0\0\200\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0", 32) = 32*

*mmap(NULL, 32, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) = 0x729412a49000*

*link("/dev/shm/sem.7rScGO", "/dev/shm/sem.semaphore\_read") = 0*

*fstat(3, {st\_mode=S\_IFREG|0664, st\_size=32, ...}) = 0*

*unlink("/dev/shm/sem.7rScGO") = 0*

*close(3) = 0*

*openat(AT\_FDCWD, "/dev/shm/virtual\_memory", O\_RDWR|O\_CREAT|O\_NOFOLLOW|O\_CLOEXEC, 0666) = 3*

*ftruncate(3, 10240) = 0*

*openat(AT\_FDCWD, "a", O\_RDONLY) = 4*

*clone(child\_stack=NULL, flags=CLONE\_CHILD\_CLEARTID|CLONE\_CHILD\_SETTID|SIGCHLD, child\_tidptr=0x729412a39a10) = 6481*

*mmap(NULL, 10240, PROT\_READ|PROT\_WRITE, MAP\_SHARED, 3, 0) = 0x729412a46000*

*futex(0x729412a49000, FUTEX\_WAIT\_BITSET|FUTEX\_CLOCK\_REALTIME, 0, NULL, FUTEX\_BITSET\_MATCH\_ANY) = 0*

*fstat(1, {st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0), ...}) = 0*

*write(1, "26.80\n", 6) = 6*

*futex(0x729412a4a000, FUTEX\_WAKE, 1) = 1*

*futex(0x729412a49000, FUTEX\_WAIT\_BITSET|FUTEX\_CLOCK\_REALTIME, 0, NULL, FUTEX\_BITSET\_MATCH\_ANY) = 0*

*write(1, "0.00\n", 5) = 5*

*futex(0x729412a4a000, FUTEX\_WAKE, 1) = 1*

*write(1, "7.51\n", 5) = 5*

*futex(0x729412a4a000, FUTEX\_WAKE, 1) = 1*

*munmap(0x729412a46000, 1024) = 0*

*wait4(6481, [{WIFEXITED(s) && WEXITSTATUS(s) == 0}], 0, NULL) = 6481*

*--- SIGCHLD {si\_signo=SIGCHLD, si\_code=CLD\_EXITED, si\_pid=6481, si\_uid=1000, si\_status=0, si\_utime=0, si\_stime=0} ---*

*munmap(0x729412a49000, 32) = 0*

*munmap(0x729412a4a000, 32) = 0*

*unlink("/dev/shm/sem.semaphore\_write") = 0*

*unlink("/dev/shm/sem.semaphore\_read") = 0*

*close(3) = 0*

*unlink("/dev/shm/virtual\_memory") = 0*

*close(4) = 0*

*exit\_group(0) = ?*

*+++ exited with 0 +++*

**Комментарии к листингу:**

Если обобщить вывод, то он разделён на 3 последовательных этапа.

Сначала с помощью различных системных вызовов инициализируется программа, загружаются нужные библиотеки, устанавливаются соответствующие значения.

Затем программа начинает выполнение. Происходит вызов кода из программы — считывается название некоторого файла. Затем идёт создание семафоров, открытие файла, создание дочернего процесса и вывод значений

Последним этапом программа завершает работу: родительский процесс ждёт завершения дочернего, удаляются все семафоры, разделяемая память, освобождается отображенная память и программа выходит.

**Вывод:**

В рамках данной лабораторной работы инструмент strace был применен для детального анализа выполнения программы. Были изучены все системные вызовы, их параметры и возвращаемые значения. В процессе работы удалось глубже понять логику программы, а также убедиться в отсутствии критических ошибок, что свидетельствует о её стабильной и корректной работе.