

**Московский авиационный институт (национальный
исследовательский университет)**

Институт информационных технологий и прикладной математики
«Кафедра вычислительной математики и программирования»

**Лабораторная работа по предмету «Операционные
системы» №8**

Студент: Брюханов З. Д.
Преподаватель: Миронов Е. С.
Группа: М8О-207Б-22
Дата: 09.12.2023
Оценка:
Подпись:

Москва 2023 г.

Оглавление

Цель работы	3
Постановка задачи	3
Общие сведения о программе	4
Пример использования strace	4
Вывод	7

Цель работы

Приобретение практических навыков диагностики работы программного обеспечения.

Постановка задачи

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

По итогам выполнения всех лабораторных работ отчет по данной ЛР должен содержать краткую сводку по исследованию написанных программ.

Общие сведения о программе

Strace - это утилита для отслеживания системных вызовов и сигналов, которые исполняет процесс. Она позволяет наблюдать взаимодействие процесса с операционной системой, что может быть полезным для отладки, анализа производительности и выявления проблем. Основные возможности strace включают в себя вывод информации о системных вызовах, сигналах, изменениях регистров, а также печать времени выполнения каждого системного вызова.

Пример использования strace

Действие утилиты продемонстрировано на примере лабораторной работы № 2.

```
root@882dead06576:/tmp/laba02/build# strace ./task02 1
execve("./task02", ["/task02", "1"], 0xffffdd9e0e48 /* 9 vars */) = 0
brk(NULL)                               = 0xaaab00044000
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS,
-1, 0) = 0xffffbad6a000
faccessat(AT_FDCWD, "/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=13219, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 13219, PROT_READ, MAP_PRIVATE, 3, 0) = 0xffffbad66000
close(3)                                = 0
openat(AT_FDCWD, "/lib/aarch64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\3\0\267\0\1\0\0\0\340u\2\0\0\0\0"...
, 832) = 832
newfstatat(3, "", {st_mode=S_IFREG|0755, st_size=1641496, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 1810024, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
0xffffbab7b000
mmap(0xffffbab80000, 1744488, PROT_READ|PROT_EXEC, MAP_PRIVATE|
MAP_FIXED|MAP_DENYWRITE, 3, 0) = 0xffffbab80000
munmap(0xffffbab7b000, 20480)           = 0
munmap(0xffffbad2a000, 44648)           = 0
mprotect(0xffffbad09000, 61440, PROT_NONE) = 0
mmap(0xffffbad18000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|
MAP_FIXED|MAP_DENYWRITE, 3, 0x188000) = 0xffffbad18000
mmap(0xffffbad1e000, 48744, PROT_READ|PROT_WRITE, MAP_PRIVATE|
MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0xffffbad1e000
close(3)                                = 0
set_tid_address(0xffffbad6af50)         = 294
set_robust_list(0xffffbad6af60, 24)     = 0
rseq(0xffffbad6b620, 0x20, 0, 0xd428bc00) = 0
mprotect(0xffffbad18000, 16384, PROT_READ) = 0
```

```

mprotect(0xaaaad2592000, 4096, PROT_READ) = 0
mprotect(0xffffbad6f000, 8192, PROT_READ) = 0
prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024,
rlim_max=RLIM64_INFINITY}) = 0
munmap(0xffffbad66000, 13219) = 0
openat(AT_FDCWD, "/sys/devices/system/cpu/online", O_RDONLY|O_CLOEXEC) = 3
read(3, "0-7\n", 1024) = 4
close(3) = 0
write(1, "enter the number of throws: ", 28enter the number of throws: ) = 28
read(0, 100
"100\n", 50) = 4
write(1, "enter the tour number: ", 23enter the tour number: ) = 23
read(0, 1
"1\n", 50) = 2
write(1, "enter the number of points of th"... , 48enter the number of points of the first player: )
= 48
read(0, 0
"0\n", 50) = 2
write(1, "enter the number of points of th"... , 49enter the number of points of the second
player: ) = 49
read(0, 0
"0\n", 50) = 2
write(1, "enter the number of experiments "... , 65enter the number of experiments that the
program should perform: ) = 65
read(0, 1000
"1000\n", 50) = 5
getrandom("\x3d\x1e\x7e\x49\xa7\x4f\x20\xf9", 8, GRND_NONBLOCK) = 8
brk(NULL) = 0xaaab00044000
brk(0xaaab00065000) = 0xaaab00065000
rt_sigaction(SIGRT_1, {sa_handler=0xffffbabfa700, sa_mask=[], sa_flags=SA_ONSTACK|
SA_RESTART|SA_SIGINFO}, NULL, 8) = 0
rt_sigprocmask(SIG_UNBLOCK, [RTMIN RT_1], NULL, 8) = 0
mmap(NULL, 8454144, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS|
MAP_STACK, -1, 0) = 0xffffba370000
mprotect(0xffffba380000, 8388608, PROT_READ|PROT_WRITE) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8) = 0
clone(child_stack=0xffffbab7e960, flags=CLONE_VM|CLONE_FS|CLONE_FILES|
CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|
CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[295], tls=0xffffbab7f8e0,
child_tidptr=0xffffbab7f1f0) = 295
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
futex(0xffffbab7f1f0, FUTEX_WAIT_BITSET|FUTEX_CLOCK_REALTIME, 295, NULL,
FUTEX_BITSET_MATCH_ANY) = 0
write(1, "Number of thread = ", 19Number of thread = ) = 19
write(1, "1", 11) = 1
write(1, "\n", 1
) = 1
write(1, "time spent on the game = ", 25time spent on the game = ) = 25

```

```

write(1, "19", 219)          = 2
write(1, "\n", 1
)          = 1
write(1, "The result is a draw: ", 22The result is a draw: ) = 22
write(1, "6", 16)            = 1
write(1, "\n", 1
)          = 1
write(1, "The result won first: ", 22The result won first: ) = 22
write(1, "488", 3488)        = 3
write(1, "\n", 1
)          = 1
write(1, "The result won second: ", 23The result won second: ) = 23
write(1, "506", 3506)        = 3
write(1, "\n\n", 2
)          = 2
write(1, "The chance of winning the first "..., 40The chance of winning the first player: ) = 40
write(1, "0.488000", 80.488000) = 8
write(1, "\n", 1
)          = 1
write(1, "The chance of winning the second"..., 41The chance of winning the second player: )
= 41
write(1, "0.506000", 80.506000) = 8
write(1, "\n", 1
)          = 1
exit_group(0)                = ?
+++ exited with 0 +++
root@882dead06576:/tmp/laba02/build#

```

Вывод

В ходе данной работы была изучена утилита `strace`, которая является удобным инструментом для отслеживания системных вызовов, используемых программой. Стоит заметить, что данная утилита выводит результат выполнения системных вызовов, что крайне полезно в процессе отладки программы. Данная утилита позволяет увидеть то, что происходит при запуске и выполнении программы с точки зрения операционной системы.