

Московский Авиационный Институт

(Национальный Исследовательский Университет)

Институт №8 “Компьютерные науки и прикладная математика”

Кафедра №806 “Вычислительная математика и программирование”

Лабораторная работа №6 по курсу
«Операционные системы»

Группа: М8О-209БВ-24

Студент: Махова А.Б.

Преподаватель: Миронов Е.С.

Оценка: _____

Дата: 19.12.25

Москва, 2025

Постановка задачи

Приобретение практических навыков диагностики работы программного обеспечения. При выполнении лабораторных работ по курсу ОС необходимо продемонстрировать ключевые системные вызовы, которые в них используются и то, что их использование соответствует варианту ЛР.

Общий метод и алгоритм решения

Утилита **strace** предназначена для трассировки системных вызовов, выполняемых процессами в операционной системе Linux. Системные вызовы являются основным механизмом взаимодействия пользовательских программ с ядром ОС. Изучаемая утилита отображает информацию о каждом системном вызове, включая его имя, переданные аргументы, возвращаемое значение и возможные ошибки. Работа strace основана на механизме, который позволяет одному процессу отслеживать выполнение другого. При каждом системном вызове выполнение трассируемой программы временно приостанавливается, после чего strace выводит соответствующую информацию. Использование утилиты strace позволяет анализировать поведение программ, выявлять причины ошибок, изучать взаимодействие процессов с ядром операционной системы и наглядно исследовать принципы работы ОС в рамках лабораторных работ. Перед тем, как перейти к практике, разберем опции утилиты и ее синтаксис:

\$ strace опции команда аргументы

В самом простом варианте strace запускает переданную команду с её аргументами и выводит в стандартный поток ошибок все системные вызовы команды. Давайте разберём опции утилиты, с помощью которых можно управлять её поведением:

- **-i** - выводить указатель на инструкцию во время выполнения системного вызова;
- **-k** - выводить стек вызовов для отслеживаемого процесса после каждого системного вызова;
- **-o** - выводить всю информацию о системных вызовах не в стандартный поток ошибок, а в файл;
- **-q** - не выводить сообщения о подключении о отключении от процесса;
- **-qq** - не выводить сообщения о завершении работы процесса;
- **-t** - выводить временную метку для каждого системного вызова;

- -s - указать максимальный размер выводимой строки, по умолчанию 32;
- -t - выводить время суток для каждого вызова;
- -tt - добавить микросекунды;
- -ttt - добавить микросекунды и количество секунд после начала эпохи Unix;
- -T - выводить длительность выполнения системного вызова;
- -x - выводить все не ASCII-строки в шестнадцатеричном виде;
- -xx - выводить все строки в шестнадцатеричном виде;
- -y - выводить пути для файловых дескрипторов;
- -yy - выводить информацию о протоколе для файловых дескрипторов;
- -c - подсчитывать количество ошибок, вызовов и время выполнения для каждого системного вызова;
- -O - добавить определённое количество микросекунд к счетчику времени для каждого вызова;
- -S - сортировать информацию выводимую при опции -c. Доступны поля time, calls, name и nothing. По умолчанию используется time;
- -w - суммировать время между началом и завершением системного вызова;
- -e - позволяет отфильтровать только нужные системные вызовы или события;
- -P - отслеживать только системные вызовы, которые касаются указанного пути;
- -v - позволяет выводить дополнительную информацию, такую как версии окружения, статистику и так далее;
- -b - если указанный системный вызов обнаружен, трассировка прекращается;
- -f - отслеживать также дочерние процессы, если они будут созданы;
- -ff - если задана опция -o, то для каждого дочернего процесса будет создан отдельный файл с именем имя_файла.pid.

- -I - позволяет блокировать реакцию на нажатия Ctrl+C и Ctrl+Z;
 - -E - добавляет переменную окружения для запускаемой программы;
 - -p - указывает pid процесса, к которому следует подключиться;
 - -u - запустить программу, от имени указанного пользователя.

Таким образом, утилита strace является эффективным средством изучения системных вызовов и механизмов взаимодействия пользовательских программ с ядром операционной системы.

Выводы strace

Лабораторная работа №1

vscode → /workspaces/MAI_OS/lab1/src (main) \$ strace -f ./parent

`execve("./parent", ["./parent"], 0xfffffb6a68c8 /* 35 vars */)=0 <запуск родительского процесса`

brk(NULL) = 0xaaaae1fc9000

```
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0xfffffb49b4000
```

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=15048, ...}, AT EMPTY PATH) = 0
```

mmap(NULL, 15048, PROT_READ, MAP_PRIVATE, 3, 0) = 0xfffffb49b0000

close(3)

```
openat(AT_FDCWD, "/lib/aarch64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
```

`newfstatat(3, "", {st mode=S_IFREG|0755, st size=1637400, ...}, AT_EMPTY_PATH) = 0`

```
mmap(NULL, 1805928, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0)=0xfffffb47c6000
```

```
mmap(0xfffffb47d0000, 1740392, PROT_READ|PROT_EXEC,  
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0) = 0xfffffb47d0000
```

munmap(0xffffb47c6000, 40960) = 0

munmap(0xfffffb4979000, 24168) = 0
mprotect(0xfffffb4958000, 61440, PROT_NONE) = 0
mmap(0xfffffb4967000, 24576, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x187000) = 0xfffffb4967000
mmap(0xfffffb496d000, 48744, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0xfffffb496d000
close(3) = 0
set_tid_address(0xfffffb49b4f50) = 5974
set_robust_list(0xfffffb49b4f60, 24) = 0
rseq(0xfffffb49b5620, 0x20, 0, 0xd428bc00) = 0
mprotect(0xfffffb4967000, 16384, PROT_READ) = 0
mprotect(0xaaad5661000, 4096, PROT_READ) = 0
mprotect(0xfffffb49b9000, 8192, PROT_READ) = 0
prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0
munmap(0xfffffb49b0000, 15048) = 0
write(1, "Input file: ", 12) = 12
read(0, test.txt
"test.txt\n", 1023) = 9
pipe2([3, 4], 0) = 0 < pipe
**clone(child_stack=NULL,
flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD**
strace: Process 6076 attached
, child_tidptr=0xfffffb49b4f50) = 6076
[pid 5974] close(4 <unfinished ...>
[pid 6076] set_robust_list(0xfffffb49b4f60, 24 <unfinished ...>
[pid 5974] <... close resumed> = 0
[pid 5974] read(3, <unfinished ...>
[pid 6076] <... set_robust_list resumed>) = 0
[pid 6076] close(3) = 0
[pid 6076] openat(AT_FDCWD, "test.txt", O_RDONLY) = 3
[pid 6076] dup3(3, 0, 0) = 0 < dup2(stdin)
[pid 6076] close(3) = 0
[pid 6076] dup3(4, 1, 0) = 1 < dup2(stdout)
[pid 6076] close(4) = 0

[pid 6076] execve("./child", ["child"], 0xfffffeb613658 /* 35 vars */) = 0 <запуск дочернего процесса

[pid 6076] brk(NULL) = 0xaaaaae9cb8000

```
[pid 6076] mmap(NULL, 8192, PROT_READ|PROT_WRITE,  
MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0xfffff9d513000
```

[pid 6076] faccessat(AT_FDCWD, "/etc/ld.so.preload", R_OK) = -1 ENOENT (No such file or directory)

[pid 6076] openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3

[pid 6076] newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=15048, ...}, AT_EMPTY_PATH) = 0

[pid 6076] mmap(NULL, 15048, PROT_READ, MAP_PRIVATE, 3, 0) = 0xfffff9d50f000

[pid 6076] close(3) = 0

[pid 6076] openat(AT_FDCWD, "/lib/aarch64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3

[pid 6076] newfstatat(3, "", {st_mode=S_IFREG|0755, st_size=1637400, ...}, AT_EMPTY_PATH) = 0

[pid 6076] mmap(NULL, 1805928, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0xfffff9d325000

```
[pid 6076] mmap(0xffff9d330000, 1740392, PROT_READ|PROT_EXEC,  
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0) = 0xffff9d330000
```

[pid 6076] munmap(0xfffff9d325000, 45056) = 0

[pid 6076] munmap(0xfffff9d4d9000, 20072) = 0

```
[pid 6076] mprotect(0xffff9d4b8000, 61440, PROT_NONE) = 0
```

```
[pid 6076] mmap(0xfffff9d4c7000, 24576, PROT_READ|PROT_WRITE,  
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x187000) = 0xfffff9d4c7000
```

```
[pid 6076] mmap(0xffff9d4cd000, 48744, PROT_READ|PROT_WRITE,  
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0xffff9d4cd000
```

[pid 6076] close(3) = 0

[pid 6076] set_tid_address(0xfffff9d513f50) = 6076

```
[pid 6076] set robust list(0xfffff9d513f60, 24) = 0
```

[pid 6076] rseq(0xfffff9d514620, 0x20, 0, 0xd428bc00) = 0

[pid 6076] mprotect(0xfffff9d4c7000, 16384, PROT_READ) = 0

[pid 6076] mprotect(0xaaaace2b1000, 4096, PROT_READ) = 0

[pid 6076] mprotect(0xfffff9d518000, 8192, PROT_READ) = 0

```
[pid 6076] prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0
```

[pid 6076] munmap(0xfffff9d50f000, 15048) = 0

[pid 6076] read(0, "12 47 -3 42\n8 12 46\n90 102 -90 2"..., 4095) = 43

[pid 6076] write(1, "98\n", 3 <unfinished ...>

[pid 5974] <... read resumed>"98\n", 4096) = 3

[pid 6076] <... write resumed> = 3

[pid 5974] write(1, "98\n", 3 <unfinished ...>

[pid 6076] write(1, "66\n", 398

<unfinished ...>

[pid 5974] <... write resumed> = 3

[pid 6076] <... write resumed> = 3

[pid 5974] read(3, <unfinished ...>

[pid 6076] write(1, "104\n", 4 <unfinished ...>

[pid 5974] <... read resumed>"66\n", 4096) = 3

[pid 6076] <... write resumed> = 4

[pid 5974] write(1, "66\n", 3 <unfinished ...>

[pid 6076] write(1, "102\n", 466

<unfinished ...>

[pid 5974] <... write resumed> = 3

[pid 6076] <... write resumed> = 4

[pid 5974] read(3, <unfinished ...>

[pid 6076] read(0, <unfinished ...>

[pid 5974] <... read resumed>"104\n102\n", 4096) = 8

[pid 6076] <... read resumed> "", 4095) = 0

[pid 5974] write(1, "104\n102\n", 8 <unfinished ...>

[pid 6076] exit_group(0104

102

<unfinished ...>

[pid 5974] <... write resumed> = 8

[pid 6076] <... exit_group resumed> = ?

[pid 5974] read(3, "", 4096) = 0

[pid 5974] close(3 <unfinished ...>

[pid 6076] +++ exited with 0 +++

<... close resumed> = 0

--- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=6076, si_uid=1000, si_status=0, si_utime=0, si_stime=0} ---

```
wait4(-1, NULL, 0, NULL) = 6076
```

```
exit_group(0) = ?
```

```
+++ exited with 0 +++
```

Лабораторная работа №2

```
clone(child_stack=0xffffabc0e960,  
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SY  
SVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTIDstrace: Process  
6531 attached  
, parent_tid=[6531], tls=0xffffabc0f8e0, child_tidptr=0xffffabc0f1f0) = 6531  
clone(child_stack=0xffffabc0e960,  
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SY  
SVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTIDstrace: Process  
6531 attached  
, parent_tid=[6531], tls=0xffffabc0f8e0, child_tidptr=0xffffabc0f1f0) = 6532  
clone(child_stack=0xffffabc0e960,  
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SY  
SVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTIDstrace: Process  
6531 attached  
, parent_tid=[6531], tls=0xffffabc0f8e0, child_tidptr=0xffffabc0f1f0) = 6533  
clone(child_stack=0xffffabc0e960,  
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SY  
SVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTIDstrace: Process  
6531 attached  
, parent_tid=[6531], tls=0xffffabc0f8e0, child_tidptr=0xffffabc0f1f0) = 6534  
clone(child_stack=0xffffabc0e960,  
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SY  
SVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTIDstrace: Process  
6531 attached  
, parent_tid=[6531], tls=0xffffabc0f8e0, child_tidptr=0xffffabc0f1f0) = 6535  
clone(child_stack=0xffffabc0e960,  
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SY  
SVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTIDstrace: Process  
6531 attached  
, parent_tid=[6531], tls=0xffffabc0f8e0, child_tidptr=0xffffabc0f1f0) = 6536  
clone(child_stack=0xffffabc0e960,  
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SY  
SVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTIDstrace: Process  
6531 attached  
, parent_tid=[6531], tls=0xffffabc0f8e0, child_tidptr=0xffffabc0f1f0) = 6537  
clone(child_stack=0xffffabc0e960,  
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SY  
SVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTIDstrace: Process  
6531 attached  
, parent_tid=[6531], tls=0xffffabc0f8e0, child_tidptr=0xffffabc0f1f0) = 6538  
clone(child_stack=0xffffabc0e960,  
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SY  
SVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTIDstrace: Process  
6531 attached
```



```
, parent_tid=[6531], tls=0xfffffabc0f8e0, child_tidptr=0xfffffabc0f1f0) = 6572
    clone(child_stack=0xfffffabc0e960,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SY
SVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTIDstrace: Process
6531 attached
, parent_tid=[6531], tls=0xfffffabc0f8e0, child_tidptr=0xfffffabc0f1f0) = 6573
    clone(child_stack=0xfffffabc0e960,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SY
SVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTIDstrace: Process
6531 attached
, parent_tid=[6531], tls=0xfffffabc0f8e0, child_tidptr=0xfffffabc0f1f0) = 6574
    clone(child_stack=0xfffffabc0e960,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SY
SVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTIDstrace: Process
6531 attached
, parent_tid=[6531], tls=0xfffffabc0f8e0, child_tidptr=0xfffffabc0f1f0) = 6575
```

Лабораторная работа №3

"a.txt\n", 127 = 6

clone(child_stack=NULL,
flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD,
child_tidptr=0xfffff99e38f50) = 35816

clock_nanosleep(CLOCK_REALTIME, 0, {tv_sec=0, tv_nsec=1000000}, NULL) = 0

write(1, "6\n", 26
) = 2

clock_nanosleep(CLOCK_REALTIME, 0, {tv_sec=0, tv_nsec=1000000}, NULL) = 0

write(1, "45\n", 345
) = 3

clock_nanosleep(CLOCK_REALTIME, 0, {tv_sec=0, tv_nsec=1000000}, NULL) = 0

write(1, "0\n", 20
) = 2

clock_nanosleep(CLOCK_REALTIME, 0, {tv_sec=0, tv_nsec=1000000}, NULL) = 0

clock_nanosleep(CLOCK_REALTIME, 0, {tv_sec=0, tv_nsec=1000000}, NULL) = ?
ERESTART_RESTARTBLOCK (Interrupted by signal)

--- SIGCHLD {si_signo=SIGHLD, si_code=CLD_EXITED, si_pid=35816, si_uid=1000, si_status=0, si_utime=0, si_stime=0} ---

restart_syscall(<... resuming interrupted clock_nanosleep ...>) = 0

clock_nanosleep(CLOCK_REALTIME, 0, {tv_sec=0, tv_nsec=1000000}, NULL) = 0

wait4(35816, NULL, 0, NULL) = 35816

munmap(0xfffff99e36000, 32) = 0

unlinkat(AT_FDCWD, "/dev/shm/sem.sum_semaphore", 0) = 0

munmap(0xfffff99e37000, 4096) = 0

unlinkat(AT_FDCWD, "/dev/shm/sum_sh_memory", 0) = 0

close(3) = 0

exit_group(0) = ?

+++ exited with 0 +++

Лабораторная работа №4

execve("./runtime", ["/./runtime"], 0xfffffe68c1a18 /* 35 vars */) = 0

brk(NULL) = 0xaaab1e700000


```
mmap(0xffff831d0000, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0) = 0xffff831d0000
close(3) = 0
mprotect(0xffff831d0000, 4096, PROT_READ) = 0
write(1, "1st dynamic library loaded\n", 271st dynamic library loaded
) = 27
write(1, "Now The naive algorithm is used "..., 59Now The naive algorithm is used for counting prime
numbers
) = 59
write(1, "Now BubbleSort is used to sort t"... , 41Now BubbleSort is used to sort the array
) = 41
write(1, "Your library has been changed su"... , 45Your library has been changed successfully!!!
) = 45
write(1, "\n", 1
) = 1
read(0, 1 1 10
"1 1 10\n", 1024) = 7
write(1, "\n", 1
) = 1
write(1, "The amount of prime numbers - 4\n", 33The amount of prime numbers - 4
) = 33
read(0, 2 5 5 4 3 2 1
"2 5 5 4 3 2 1\n", 1024) = 14
write(1, "Sorted array - 1 2 3 4 5 \n", 26Sorted array - 1 2 3 4 5
) = 26
write(1, "\n", 1
) = 1
read(0, -1
"-1\n", 1024) = 3
munmap(0xffff831c0000, 69672) = 0
write(1, "The program successfully finished", 33The program successfully finished
) = 33
lseek(0, -1, SEEK_CUR) = -1 ESPIPE (Illegal seek)
```

```
exit_group(0)          = ?
```

```
+++ exited with 0 +++
```

Вывод

В ходе выполнения лабораторных работ с использованием утилиты **strace** были изучены основные механизмы взаимодействия пользовательских программ с ядром операционной системы Linux. На практике были проанализированы системные вызовы, связанные с созданием и завершением процессов, межпроцессным взаимодействием, перенаправлением потоков ввода-вывода, управлением памятью и обработкой сигналов.