

Московский Авиационный Институт
(Национальный Исследовательский Университет)
Факультет информационных технологий и прикладной математики
Кафедра вычислительной математики и программирования

Лабораторная работа №1 по курсу
«Операционные системы»

Тема работы
“Использование утилиты strace”

Студент: Прохоров Данила
Михайлович

Группа: М8О-208Б-20

Вариант: -

Преподаватель: Миронов Евгений Сергеевич

Оценка: _____

Дата: _____

Подпись: _____

Москва, 2021

Содержание

1. Репозиторий
2. Постановка задачи
3. Демонстрация работы с утилитой `strace` и подробное объяснение каждого системного вызова
4. Выводы

Репозиторий

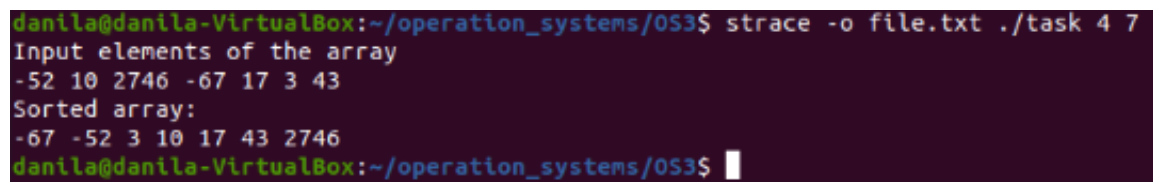
<https://github.com/dmprokhorov>

Постановка задачи

Задача: подробно рассказать о каждом системном вызове, отображенном утилитой strace. Разбор системных вызовов осуществляется на примере третьей лабораторной работы курса “Операционные системы”.

Демонстрация работы с утилитой strace и подробное объяснение каждого системного вызова

Запуск программы:



```
danila@danila-VirtualBox:~/operation_systems/OS3$ strace -o file.txt ./task 4 7
Input elements of the array
-52 10 2746 -67 17 3 43
Sorted array:
-67 -52 3 10 17 43 2746
danila@danila-VirtualBox:~/operation_systems/OS3$
```

Исходный код strace:

```
execve("./task", [".task", "4", "7"], 0x7ffdb439900 /* 49 vars */) = 0

brk(NULL)                               = 0x5614331b7000
arch_prctl(0x3001 /* ARCH_??? */, 0x7fffe6e665b0) = -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
fstat(3, {st_mode=S_IFREG|0644, st_size=74405, ...}) = 0
mmap(NULL, 74405, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f698880e000
close(3)                                = 0
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libpthread.so.0", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\220\201\0\0\0\0"... , 832)
= 832
pread64(3,
"\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\345Ga\367\265T\320\374\301V)Yf]\223\337"... , 68, 824)
= 68
fstat(3, {st_mode=S_IFREG|0755, st_size=157224, ...}) = 0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
0x7f698880c000
pread64(3,
"\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\345Ga\367\265T\320\374\301V)Yf]\223\337"... , 68, 824)
= 68
mmap(NULL, 140408, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f69887e9000
```

```

mmap(0x7f69887f0000, 69632, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x7000) = 0x7f69887f0000
mmap(0x7f6988801000, 20480, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x18000) = 0x7f6988801000
mmap(0x7f6988806000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x1c000) = 0x7f6988806000
mmap(0x7f6988808000, 13432, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f6988808000
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\360q\2\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\2\0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0", 32,
848) = 32
pread64(3,
"\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\t\233\222%\274\260\320\31\331\326\10\204\276X>\263"..
., 68, 880) = 68
fstat(3, {st_mode=S_IFREG|0755, st_size=2029224, ...}) = 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
pread64(3, "\4\0\0\0\2\0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0", 32,
848) = 32
pread64(3,
"\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\t\233\222%\274\260\320\31\331\326\10\204\276X>\263"..
., 68, 880) = 68
mmap(NULL, 2036952, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f69885f7000
mprotect(0x7f698861c000, 1847296, PROT_NONE) = 0
mmap(0x7f698861c000, 1540096, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x25000) = 0x7f698861c000
mmap(0x7f6988794000, 303104, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x19d000) = 0x7f6988794000
mmap(0x7f69887df000, 24576, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1e7000) = 0x7f69887df000
mmap(0x7f69887e5000, 13528, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f69887e5000
close(3) = 0
mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
0x7f69885f4000
arch_prctl(ARCH_SET_FS, 0x7f69885f4740) = 0
mprotect(0x7f69887df000, 12288, PROT_READ) = 0
mprotect(0x7f6988806000, 4096, PROT_READ) = 0
mprotect(0x561432234000, 4096, PROT_READ) = 0
mprotect(0x7f698884e000, 4096, PROT_READ) = 0
munmap(0x7f698880e000, 74405) = 0
set_tid_address(0x7f69885f4a10) = 3846
set_robust_list(0x7f69885f4a20, 24) = 0

```

```

rt_sigaction(SIGRTMIN, {sa_handler=0x7f69887f0bf0, sa_mask=[],
sa_flags=SA_RESTORER|SA_SIGINFO, sa_restorer=0x7f69887fe3c0}, NULL, 8) = 0
rt_sigaction(SIGRT_1, {sa_handler=0x7f69887f0c90, sa_mask=[],
sa_flags=SA_RESTORER|SA_RESTART|SA_SIGINFO, sa_restorer=0x7f69887fe3c0}, NULL, 8) = 0
rt_sigprocmask(SIG_UNBLOCK, [RTMIN RT_1], NULL, 8) = 0
prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0
fstat(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0), ...}) = 0
brk(NULL) = 0x5614331b7000
brk(0x5614331d8000) = 0x5614331d8000
write(1, "Input elements of the array\n", 28) = 28
fstat(0, {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0), ...}) = 0
read(0, "-52 10 2746 -67 17 3 43\n", 1024) = 24
mmap(NULL, 8392704, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0) =
0x7f6987df3000
mprotect(0x7f6987df4000, 8388608, PROT_READ|PROT_WRITE) = 0
clone(child_stack=0x7f69885f2fb0,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SE
TTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[3850], tls=0x7f69885f3700,
child_tidptr=0x7f69885f39d0) = 3850
futex(0x7f69885f39d0, FUTEX_WAIT, 3850, NULL) = 0
write(1, "Sorted array:\n", 14) = 14
write(1, "-67 -52 3 10 17 43 2746 \n", 25) = 25
lseek(0, -1, SEEK_CUR) = -1 ESPIPE (Недопустимая операция смещения)
exit_group(0) = ?
+++ exited with 0 +++

```

Разберем подробнее:

`execve(...)` - запускает программу

`brk(...)` - устанавливает конец сегмента данных в значение `NULL`

`access(...)` - проверяет права доступа к файлу, возвращает -1 как код ошибки

`openat(...)` - открывает файл, имеет в качестве возвращаемого значения файловый дескриптор

`fstat(...)` - собирает информацию из файла

`mmap(...)` - отображает файл на память

`mprotect(...)` - контролирует доступ к памяти

`close(...)` - закрывает файловый дескриптор

`read(...)` - считывает из файлового дескриптора

`arch_prctl(...)` - устанавливает специфичное для архитектуры значение ядра

`munmap(...)` - освобождает память, отведенную для отображения файла

`write(...)` - пишет в консоль

`clone(...)` – создаёт дочерний процесс

`futex(...)` – ожидание, пока не произойдёт какое-то событие.

set_tid_address(...) – при запуске потока с помощью clone с флагом CLONE_CHILD_SETTID в значение set_child_tid устанавливается равным аргументу системного вызова ctid

set_robust_list(...) – возвращает начало списка надёжных фьютексов нити (потока)

rt_sigaction(...) - изменит выполняемого процессом действия при получении определённого сигнала

rt_sigprocmask(...) - выборка и/или изменение маски сигналов вызывающей нити

prlimit64(...) – устанавливает ограничение ресурса для процесса

pread64(...) – читает из файлового дескриптора

getcwd(...) - получить текущую рабочую директорию

Выводы

В данной лабораторной работе я углублённо разобрал strace и убедился, что это очень полезная утилита для просмотра системных вызовов.