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

Лабораторная работа №1 по курсу
«Операционные системы»

Диагностика программного обеспечения

Студент: Косогоров Владислав Валерьевич
Группа: М80 – 206Б-18
Вариант: 1
Преподаватель: Соколов Андрей Алексеевич
Оценка: _____
Дата: _____
Подпись: _____

Москва, 2019

Постановка задачи

Цель работы — приобретение практических навыков диагностики работы программного обеспечения.

Strace

Strace показывает все системные вызовы программы, которые она отправляет к системе во время выполнения, а также их параметры и результат выполнения.

Основные ключи strace:

- **-i** — выводить указатель на инструкцию во время выполнения системного вызова
- **-o** — выводить всю информацию о системных вызовах не в стандартный поток ошибок, а в файл
- **-T** — выводить длительность выполнения каждого системного вызова
- **-l** — позволяет блокировать реакцию нажатия Ctrl+C и Ctrl+Z
- **-f** — отслеживать дочерние процессы и создаваемые потоки

Общий метод и алгоритм решения

Проведём диагностику лабораторной работы номер 4.

Код программы

main.c:

```
#include <stdio.h>
#include <sys/mman.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <stdlib.h>
#include <sys/types.h>
#include <unistd.h>
#include <ctype.h>
#include <sys/wait.h>


const int ARG_MAX = 2097152 + 1;

int main(int argc, char** argv) {
    pid_t pid;
    int rv, fd;
```

```
    char* filename = "memfile";

    if ((fd = shm_open(filename, O_RDWR | O_CREAT | O_TRUNC, S_IRUSR |
S_IWUSR)) == -1) {
        perror("shm_open error");
        exit(-1);
    }

    if (ftruncate(fd, ARG_MAX * 2) == -1) {
        perror("failed to truncate");
        exit(-1);
    }

    char* memory = mmap(NULL, ARG_MAX * 2, PROT_READ |
PROT_WRITE, MAP_SHARED, fd, 0);

    if (memory == MAP_FAILED) {
        perror("mapping error");
        fprintf(stderr, "%p", memory);
        exit(-1);
    }

    if ((pid = fork()) < 0) {
        perror("fork error");
        exit(-1);

    } else if (pid == 0) {

        dup2(fd, STDOUT_FILENO);

        rv = execvp(argv[1], argv + 1);
        if (rv) {
            perror("exec error");
        }
        exit(rv);

    } else if (pid > 0) {
        waitpid(pid, &rv, 0);

        for (int i = 0; memory[i] != '\0'; ++i) {
            if (islower(memory[i])) {
                putchar(toupper(memory[i]));
```

```
            } else {
                putchar(memory[i]);
            }
        }

        exit(WEXITSTATUS(rv));
    }

    if (munmap(memory, ARG_MAX * 2)) {
        perror("munmap error");
        exit(-1);
    }

    return 0;
}
```

Демонстрация работы программы

…/prog_3_sem/os/lab4 ż strace -T -i ./a.out cat main.c | tail

```
[00007f30c6263e37] execve("./a.out", ["./a.out", "cat", "main.c"], 0x7f
ff5f2a53d0 /* 79 vars */) = 0 <0.000686>
[00007f4aaaad8ec9] brk(NULL)             = 0x559b93e30000 <0.000034>
[00007f4aaaacc7de] access("/etc/ld.so.nohwcap", F_OK) = -1 ENOENT (No s
uch file or directory) <0.000039>
[00007f4aaaad9e27] access("/etc/ld.so.preload", R_OK) = -1 ENOENT (No s
uch file or directory) <0.000037>
[00007f4aaaad9cdd] openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|
O_CLOE
XEC) = 3 <0.000039>
[00007f4aaaad9c43] fstat(3, {st_mode=S_IFREG|0644, st_size=155179, ...}
) = 0 <0.000025>
[00007f4aaaad9f43] mmap(NULL, 155179, PROT_READ, MAP_PRIVATE, 3, 0)
= 0
x7f4aaacbe000 <0.000032>
[00007f4aaaad9ed7] close(3)              = 0 <0.000026>
[00007f4aaaad5139] access("/etc/ld.so.nohwcap", F_OK) = -1 ENOENT (No s
uch file or directory) <0.000033>
[00007f4aaaad9cdd] openat(AT_FDCWD, "/lib/x86_64-linux-gnu/librt.so.1",
O_RDONLY|O_CLOEXEC) = 3 <0.000044>
[00007f4aaaad9da4] 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"..., 832) = 832 <0.000032>
[00007f4aaaad9c43] fstat(3, {st_mode=S_IFREG|0644, st_size=31680, ...})
= 0 <0.000029>
```

```
[00007f4aaaad9f43] mmap(NULL, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|M
AP_ANONYMOUS, -1, 0) = 0x7f4aaacbc000 <0.000035>
[00007f4aaaad9f43] mmap(NULL, 2128864, PROT_READ|PROT_EXEC,
MAP_PRIVATE
|MAP_DENYWRITE, 3, 0) = 0x7f4aaa8b5000 <0.000038>
[00007f4aaaad9ff7] mprotect(0x7f4aaa8bc000, 2093056, PROT_NONE) = 0 <0.
000045>
[00007f4aaaad9f43] mmap(0x7f4aaaabb000, 8192, PROT_READ|PROT_WRITE,
MAP
_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x6000) = 0x7f4aaaabb000
<0.000046
>
[00007f4aaaad9ed7] close(3)             = 0 <0.000027>
[00007f4aaaad5139] access("/etc/ld.so.nohwcap", F_OK) = -1 ENOENT (No s
uch file or directory) <0.000040>
[00007f4aaaad9cdd] openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6",
O_RDONLY|O_CLOEXEC) = 3 <0.000045>
[00007f4aaaad9da4] read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\
0\0\260\34\2\0\0\0\0\0"..., 832) = 832 <0.000034>
[00007f4aaaad9c43] fstat(3, {st_mode=S_IFREG|0755, st_size=2030544, ...
}) = 0 <0.000031>
[00007f4aaaad9f43] mmap(NULL, 4131552, PROT_READ|PROT_EXEC,
MAP_PRIVATE
|MAP_DENYWRITE, 3, 0) = 0x7f4aaa4c4000 <0.000041>
[00007f4aaaad9ff7] mprotect(0x7f4aaa6ab000, 2097152, PROT_NONE) = 0 <0.
000044>
[00007f4aaaad9f43] mmap(0x7f4aaa8ab000, 24576, PROT_READ|
PROT_WRITE, MA
P_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1e7000) = 0x7f4aaa8ab000
<0.000
043>
[00007f4aaaad9f43] mmap(0x7f4aaa8b1000, 15072, PROT_READ|
PROT_WRITE, MA
P_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f4aaa8b1000
<0.000033>
[00007f4aaaad9ed7] close(3)             = 0 <0.000031>
[00007f4aaaad5139] access("/etc/ld.so.nohwcap", F_OK) = -1 ENOENT (No s
uch file or directory) <0.000033>
[00007f4aaaad9cdd] openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libpthread.s
o.0", O_RDONLY|O_CLOEXEC) = 3 <0.000037>
[00007f4aaaad9da4] read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\3\0>\0\1\0\
```

```
0\0000b\0\0\0\0\0\0"..., 832) = 832 <0.000030>
[00007f4aaaad9c43] fstat(3, {st_mode=S_IFREG|0755, st_size=144976, ...}
) = 0 <0.000031>
[00007f4aaaad9f43] mmap(NULL, 2221184, PROT_READ|PROT_EXEC,
MAP_PRIVATE
|MAP_DENYWRITE, 3, 0) = 0x7f4aaa2a5000 <0.000068>
[00007f4aaaad9ff7] mprotect(0x7f4aaa2bf000, 2093056, PROT_NONE) = 0 <0.
000063>
[00007f4aaaad9f43] mmap(0x7f4aaa4be000, 8192, PROT_READ|PROT_WRITE,
MAP
_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x19000) = 0x7f4aaa4be000
<0.00006
1>
[00007f4aaaad9f43] mmap(0x7f4aaa4c0000, 13440, PROT_READ|
PROT_WRITE, MA
P_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f4aaa4c0000
<0.000033>
[00007f4aaaad9ed7] close(3)              = 0 <0.000033>
[00007f4aaaad9f43] mmap(NULL, 12288, PROT_READ|PROT_WRITE,
MAP_PRIVATE|
MAP_ANONYMOUS, -1, 0) = 0x7f4aaacb9000 <0.000032>
[00007f4aaaabe024] arch_prctl(ARCH_SET_FS, 0x7f4aaacb9740) = 0 <0.00001
8>
[00007f4aaaad9ff7] mprotect(0x7f4aaa8ab000, 16384, PROT_READ) = 0 <0.00
0042>
[00007f4aaaad9ff7] mprotect(0x7f4aaa4be000, 4096, PROT_READ) = 0 <0.000
036>
[00007f4aaaad9ff7] mprotect(0x7f4aaaabb000, 4096, PROT_READ) = 0 <0.000
043>
[00007f4aaaad9ff7] mprotect(0x559b93c48000, 4096, PROT_READ) = 0 <0.000
042>
[00007f4aaaad9ff7] mprotect(0x7f4aaace4000, 4096, PROT_READ) = 0 <0.000
041>
[00007f4aaaad9fd7] munmap(0x7f4aaacbe000, 155179) = 0 <0.000053>
[00007f4aaa2aaeb5] set_tid_address(0x7f4aaacb9a10) = 2392 <0.000033>
[00007f4aaa2aaf17] set_robust_list(0x7f4aaacb9a20, 24) = 0 <0.000033>
[00007f4aaa2b795d] rt_sigaction(SIGRTMIN, {sa_handler=0x7f4aaa2aacb0, s
a_mask=[], sa_flags=SA_RESTORER|SA_SIGINFO,
sa_restorer=0x7f4aaa2b7890}
, NULL, 8) = 0 <0.000030>
[00007f4aaa2b795d] rt_sigaction(SIGRT_1, {sa_handler=0x7f4aaa2aad50, sa
_mask=[], sa_flags=SA_RESTORER|SA_RESTART|SA_SIGINFO,
```

```
sa_restorer=0x7f4
aaa2b7890}, NULL, 8) = 0 <0.000034>
[00007f4aaa2aaff3] rt_sigprocmask(SIG_UNBLOCK, [RTMIN RT_1], NULL, 8)
=
0 <0.000039>
[00007f4aaa5d9fa0] prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024
, rlim_max=RLIM64_INFINITY}) = 0 <0.000029>
[00007f4aaa5d9fa0] prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024
, rlim_max=RLIM64_INFINITY}) = 0 <0.000026>
[00007f4aaa5d3977] statfs("/dev/shm/", {f_type=TMPFS_MAGIC, f_bsize=409
6, f_blocks=1005974, f_bfree=980459, f_bavail=980459, f_files=1005974,
f_ffree=1005817, f_fsid={val=[0, 0]}, f_namelen=255, f_frsize=4096, f_f
lags=ST_VALID|ST_NOSUID|ST_NODEV}) = 0 <0.000045>
[00007f4aaa2b484e] futex(0x7f4aaa4c3370, FUTEX_WAKE_PRIVATE,
2147483647
) = 0 <0.000028>
[00007f4aaa5d3c8e] openat(AT_FDCWD, "/dev/shm/memfile", O_RDWR|
O_CREAT|
O_TRUNC|O_NOFOLLOW|O_CLOEXEC, 0600) = 3 <0.000058>
[00007f4aaa5dcc97] ftruncate(3, 4194304) = 0 <0.000032>
[00007f4aaa5dfa13] mmap(NULL, 4194304, PROT_READ|PROT_WRITE,
MAP_SHARED
, 3, 0) = 0x7f4aa9ea5000 <0.000039>
[00007f4aaa5a8b1c] clone(child_stack=NULL,
flags=CLONE_CHILD_CLEARTID|C
LONE_CHILD_SETTID|SIGCHLD, child_tidptr=0x7f4aaacb9a10) = 2393
<0.00015
3>
[00007f4aaa5a8687] wait4(2393, [{WIFEXITED(s) && WEXITSTATUS(s) ==
0}],
0, NULL) = 2393 <0.001816>
[00007f4aaa5a8687] --- SIGCHLD {si_signo=SIGCHLD,
si_code=CLD_EXITED, s
i_pid=2393, si_uid=1000, si_status=0, si_utime=0, si_stime=0} ---
[00007f4aaa5d37c3] fstat(1, {st_mode=S_IFIFO|0600, st_size=0, ...}) = 0
<0.000028>
[00007f4aaa5da4b9] brk(NULL)            = 0x559b93e30000 <0.000026>
[00007f4aaa5da4b9] brk(0x559b93e51000)  = 0x559b93e51000 <0.000031>
[00007f4aaa5d4154] write(1, "#INCLUDE <STDIO.H>\n#INCLUDE <SYS"...,
145
2) = 1452 <0.000039>
[00007f4aaa5a8e06] exit_group(0)        = ?
```

```
[????????????????] +++ exited with 0 +++
      EXIT(WEXITSTATUS(RV));
   }

   IF (MUNMAP(MEMORY, ARG_MAX * 2)) {
      PERROR("MUNMAP ERROR");
      EXIT(-1);
   }

   RETURN 0;
}
```

Вывод

В результате данной лабораторной работы я на практике познакомился с возможностями утилиты strace, продиагностировав ранее выполненную лабораторную работу.