Московский Авиационный Институт

(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

**Лабораторная работа**

**по курсу «Операционные системы»**

**III Семестр**

**Задание 4**

**Вариант 16**

|  |  |
| --- | --- |
| Студент: | Жерлыгин М.А. |
| Группа: | М80-208Б-18 |
| Преподаватель: | Миронов Е.С |
|  |  |
| Оценка: |  |
| Дата: |  |

Москва 2019

**1. Описание задания**

Составить и отладить программу на языке Си, осуществляющую работу с процессами и взаимодействие между ними в одной из двух операционных систем. В результате работы программа (основной процесс) должен создать для решение задачи один или несколько дочерних процессов. Взаимодействие между процессами осуществляется через системные сигналы/события и/или каналы (pipe).

Необходимо обрабатывать системные ошибки, которые могут возникнуть в результате работы.

Вариант 16: На вход программе подается команда интерпретатора команд и имя файла. Программа должна перенаправить стандартный ввод команды с этого файла и вывести результат команды в стандартный выходной поток. Использование операций write и printf запрещено.

**2. Код программы:**

#include <sys/types.h>

#include <unistd.h>

#include <stdio.h>

#include <stdlib.h>

#include <fcntl.h>

#include <sys/wait.h>

#include<sys/mman.h>

#include <sys/stat.h>

#include <string.h>

int main(int argc, char\* argv[]){

if (argc != 2) {

fprintf(stderr, "Invalid input\n");

exit(2);

}

pid\_t pid;

int status;

struct stat statbuf;

int fd = open(argv[1], O\_RDWR);

if (fd == -1){

fprintf(stderr, "Cant open file\n");

}

if (fstat(fd, &statbuf) < 0 ) {

fprintf(stderr,"Fstat error\n");

} else fstat(fd, &statbuf);

char\* filemap = (char\*)mmap(0, statbuf.st\_size, PROT\_READ, MAP\_SHARED, fd, 0);

if (filemap == MAP\_FAILED){

fprintf(stderr,"Error mmap");

}

if(close(fd) == -1){

fprintf(stderr, "Cant close file\n");

} else close(fd);

pid = fork();

if(pid < 0) {

fprintf(stderr, "Prosses not created\n");

exit(1);

} else if(pid == 0){

char a[255] = {0};

char b[255] = {0};

int n = 0;

while(filemap[n] != '\n' && filemap[n] != EOF) {

if(n == 256) {

fprintf(stderr, "Incorrect command\n");

munmap(filemap, statbuf.st\_size);

exit(4);

}

a[n]= filemap[n];

n++;

}

n++;

int k = 0;

while(filemap[n] != '\n' && filemap[n] != EOF) {

if(k == 256) {

fprintf(stderr, "Incorrect file name\n");

munmap(filemap, statbuf.st\_size);

exit(4);

}

b[k]= filemap[n];

n++;

k++;

}

if(execlp(a, a, b, (char\*)NULL) == -1){

fprintf(stderr, "Execlp error\n");

} else execlp(a, a, b, (char\*)NULL);

}else if(pid > 0){

if (waitpid(pid, &status, 0) == -1){

fprintf(stderr, "Waitpid error\n");

} else waitpid(pid, &status, 0);

munmap(filemap, statbuf.st\_size);

}

return 0;

}

**3. Протокол**

mmaxim2710@DESKTOP-RDPBU3D:~/OC2$ gcc -g lab4.c -o main

mmaxim2710@DESKTOP-RDPBU3D:~/OC2$ ./main in\_test

123123123123123

[mmaxim2710@DESKTOP-RDPBU3D](mailto:mmaxim2710@DESKTOP-RDPBU3D):~/OC2$

**4. Объяснение результата работы программы**

Программа получает название файла, в котором содержатся команда для выполнение и название файла, к которому применяется команда. После данные файла отображатся в память, создается дочерний процесс, который считывает содержимое отображение и выполняет команду.

**5. Набор тестов**

1) Входные данные:

./main in\_test

Выходные данные:

123123123123123

2) Входные данные:

./main in\_test in\_test1

Выходные данные:

Invalid input

**6. Strace**

execve("./main", ["./main", "in\_test"], 0x7fffffc06d58 /\* 19 vars \*/) = 0

brk(NULL) = 0x7fffc4b49000

access("/etc/ld.so.nohwcap", F\_OK) = -1 ENOENT (No such file or directory)

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=54618, ...}) = 0

mmap(NULL, 54618, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7f68290a8000

close(3) = 0

access("/etc/ld.so.nohwcap", F\_OK) = -1 ENOENT (No such file or directory)

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\260\34\2\0\0\0\0\0"..., 832) = 832

fstat(3, {st\_mode=S\_IFREG|0755, st\_size=2030544, ...}) = 0

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f68290a0000

mmap(NULL, 4131552, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f6828a00000

mprotect(0x7f6828be7000, 2097152, PROT\_NONE) = 0

mmap(0x7f6828de7000, 24576, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1e7000) = 0x7f6828de7000

mmap(0x7f6828ded000, 15072, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7f6828ded000

close(3) = 0

arch\_prctl(ARCH\_SET\_FS, 0x7f68290a14c0) = 0

mprotect(0x7f6828de7000, 16384, PROT\_READ) = 0

mprotect(0x7f6829401000, 4096, PROT\_READ) = 0

mprotect(0x7f6829027000, 4096, PROT\_READ) = 0

munmap(0x7f68290a8000, 54618) = 0

openat(AT\_FDCWD, "in\_test", O\_RDWR) = 3

fstat(3, {st\_mode=S\_IFREG|0666, st\_size=8, ...}) = 0

fstat(3, {st\_mode=S\_IFREG|0666, st\_size=8, ...}) = 0

mmap(NULL, 8, PROT\_READ, MAP\_SHARED, 3, 0) = 0x7f68290b5000

close(3) = 0

close(3) = -1 EBADF (Bad file descriptor)

clone(child\_stack=NULL, flags=CLONE\_CHILD\_CLEARTID|CLONE\_CHILD\_SETTID|SIGCHLD, child\_tidptr=0x7f68290a1790) = 53

wait4(53, [{WIFEXITED(s) && WEXITSTATUS(s) == 0}], 0, NULL) = 53

--- SIGCHLD {si\_signo=SIGCHLD, si\_code=CLD\_EXITED, si\_pid=53, si\_uid=1000, si\_status=0, si\_utime=0, si\_stime=0} ---

wait4(53, 0x7fffcc1d70a4, 0, NULL) = -1 ECHILD (No child processes)

munmap(0x7f68290b5000, 8) = 0

exit\_group(0) = ?

+++ exited with 0 +++

**7. Вывод**

Проделав данную работу я пришёл к выводу, что преимуществом использования отображения является меньшая, по сравнению с чтением/записью, нагрузка на операционную систему. При этом чтение данных из этих адресов фактически приводит к чтению данных из отображенного файла, а запись данных по этим адресам приводит к записи этих данных в файл.