Московский Авиационный Институт

(Национальный Исследовательский Университет)

Институт №8 “Компьютерные науки и прикладная математика”

Кафедра №806 “Вычислительная математика и программирование”

Лабораторная работа №1 по курсу

«Операционные системы»

Группа: М8О-213Б-23

Студент: Барменков А.С.

Преподаватель: Бахарев В.Д.

Оценка: _____

Дата: 05.10.24

Москва, 2024

Постановка задачи

Вариант 19.

Родительский процесс создает два дочерних процесса. Первой строкой пользователь в консоль родительского процесса вводит имя файла, которое будет использовано для открытия File с таким именем на запись для child1. Аналогично для второй строки и процесса child2. Родительский и дочерний процесс должны быть представлены разными программами.
Родительский процесс принимает от пользователя строки произвольной длины и пересылает их в pipe1 или в pipe2 в зависимости от правила фильтрации. Процесс child1 и child2 производят работу над строками. Процессы пишут результаты своей работы в стандартный вывод.

Правило фильтрации: с вероятностью 80% строки отправляются в pipe1, иначе в pipe2.
Дочерние процессы удаляют все гласные из строк.

Общий метод и алгоритм решения

Использованные системные вызовы:

- pipe() - Создает неименованный канал (pipe) для обмена данными между процессами.
- fork() - Создает новый процесс путем копирования текущего (родительского) процесса.
- close() - Закрывает файловый дескриптор, освобождая связанные с ним системные ресурсы.
- dup2() - Дублирует один файловый дескриптор в другой.
- execlp() - Выполняет указанную программу, заменяя текущий процесс новой программой.
- wait() - Ожидает завершения процесса.
- getline() - Читает строку из потока ввода (например, стандартного ввода) и выделяет для неё память.
- perror() - Выводит сообщение об ошибке, соответствующее глобальной переменной `errno`, на стандартный поток ошибок.
- exit() - Завершает выполнение программы с указанным кодом выхода.
- write() - Пишет данные из буфера в файл или файловый дескриптор, например, в канал.
- srand() - Инициализирует генератор случайных чисел с указанным начальным значением.
- rand() - Возвращает случайное целое число.
- strcmp() - Сравнивает две строки. Возвращает 0, если строки идентичны.
- strcspn() - Возвращает индекс первого вхождения символа из второй строки в первой строке.

Программа начинает с того, что родительский процесс создает два канала для передачи данных между собой и двумя дочерними процессами. Эти каналы позволяют родителю отправлять строки дочерним процессам для дальнейшей обработки. Родитель затем создает дочерние процессы с помощью fork(). После этого родительский процесс ожидает пользовательский ввод и пересылает его через каналы дочерним процессам. С вероятностью 80% данные попадут в pipe1.

Дочерний процесс принимает данные и удаляет все гласные буквы. Затем дочерний процесс записывает в ранее введённый файл результат своей работы.

В итоге, программа организует параллельное выполнение двух дочерних процессов, которым через каналы передаются строки от родителя. Каждый дочерний процесс обрабатывает строки (удаляет гласные) и записывает их в свой файл. Родительский процесс решает, в какой процесс отправить строку, используя генерацию случайного числа.

Код программы

parent.c

```
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <fcntl.h>
#include <sys/wait.h>
#include <time.h>

void HandleError(const char* msg) {
    write(STDERR_FILENO, msg, strlen(msg));
    write(STDERR_FILENO, "\n", 1);
    exit(1);
}

ssize_t Getline(char **lineptr, size_t *n, int fd) {
    if (*lineptr == NULL) {
        *lineptr = malloc(128);
        *n = 128;
    }

    size_t pos = 0;
    char c;
    while (read(fd, &c, 1) == 1) {
        if (pos >= *n - 1) {
            *n *= 2;
            *lineptr = realloc(*lineptr, *n);
        }
        (*lineptr)[pos++] = c;
        if (c == '\n') {
            break;
        }
    }
```

```
    if (pos == 0) {
        return -1;
    }

    (*lineptr)[pos] = '\0';
    return pos;
}

void Print(const char* msg) {
    write(STDOUT_FILENO, msg, strlen(msg));
}

int main() {
    int pipe1[2], pipe2[2];
    pid_t child1, child2;
    char *input = NULL;
    size_t len = 0;
    ssize_t nread;
    int r = 0;

    if (pipe(pipe1) == -1 || pipe(pipe2) == -1) {
        HandleError("Pipe failed");
    }

    char *file1 = NULL, *file2 = NULL;
    size_t file_len = 0;

    Print("Введите имя файла для дочернего процесса 1: ");
    Getline(&file1, &file_len, STDIN_FILENO);
    file1[strcspn(file1, "\n")] = 0;

    Print("Введите имя файла для дочернего процесса 2: ");
    Getline(&file2, &file_len, STDIN_FILENO);
    file2[strcspn(file2, "\n")] = 0;

    if ((child1 = fork()) == 0) {
        close(pipe1[1]);
        close(pipe2[0]);
        close(pipe2[1]);

        dup2(pipe1[0], STDIN_FILENO);
        close(pipe1[0]);

        execlp("./child1", "./child1", file1, NULL);
        HandleError("execve for child1 failed");
    }

    if ((child2 = fork()) == 0) {
        close(pipe2[1]);
        close(pipe1[0]);
```

```
        close(pipe1[1]);

        dup2(pipe2[0], STDIN_FILENO);
        close(pipe2[0]);

        execlp("./child2", "./child2", file2, NULL);
        HandleError("execve for child2 failed");
    }

    close(pipe1[0]);
    close(pipe2[0]);

    srand(time(NULL));

    while (1) {
        Print("Введите строку (или 'exit' для завершения): ");
        nread = Getline(&input, &len, STDIN_FILENO);
        input[strcspn(input, "\n")] = 0;

        if (strcmp(input, "exit") == 0) {
            break;
        }

        r = rand() % 5 + 1;

        if (r == 3) {
            write(pipe2[1], input, nread);
        } else {
            write(pipe1[1], input, nread);
        }
    }

    close(pipe1[1]);
    close(pipe2[1]);

    wait(NULL);
    wait(NULL);

    Print("Работа завершена.\n");

    free(input);
    free(file1);
    free(file2);

    return 0;
}
```

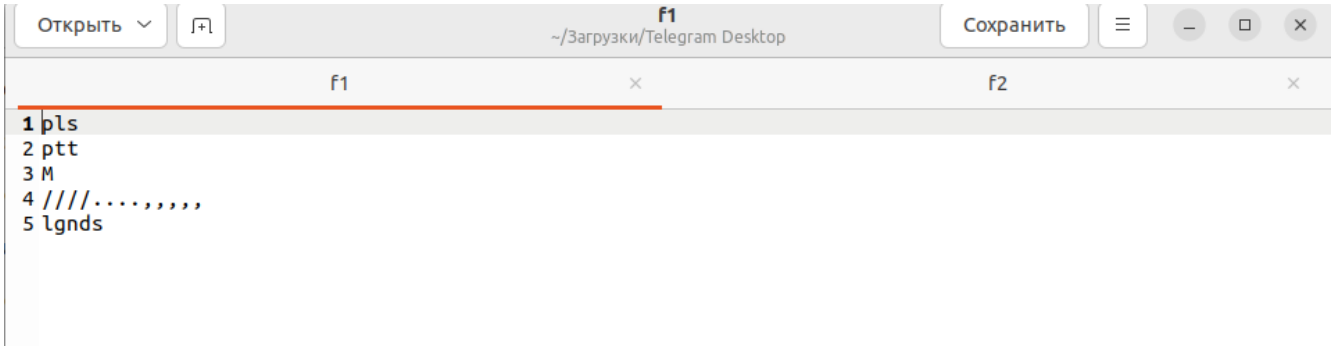
child1.c/ child2.c

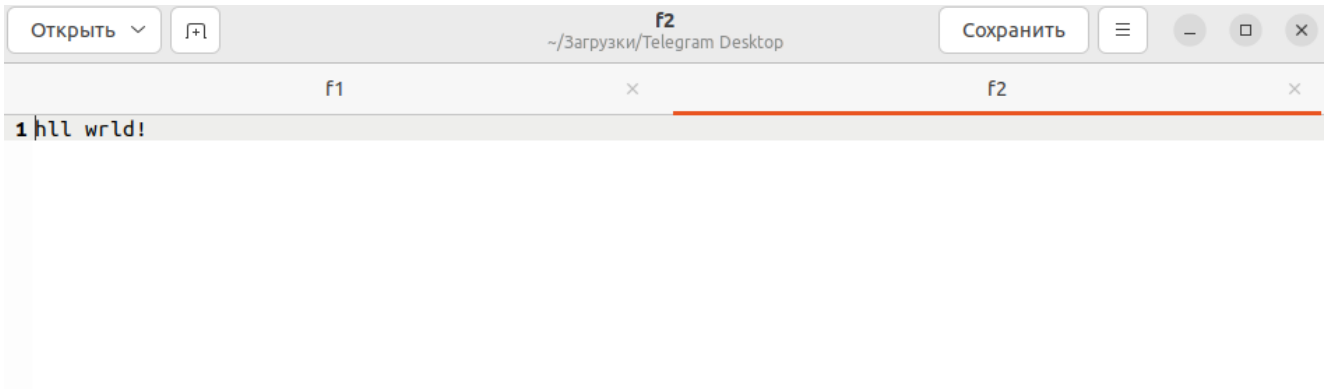
```
#include <stdlib.h>
#include <string.h>
#include <fcntl.h>
#include <unistd.h>

void HandleError(const char* msg) {
    write(STDERR_FILENO, msg, strlen(msg));
    write(STDERR_FILENO, "\n", 1);
    exit(1);
}

void Print(const char* msg) {
    write(STDOUT_FILENO, msg, strlen(msg));
}

void RemoveVowels(char* str) {
    char* p = str;
    char* q = str;
    while (*p) {
        if (!strchr("AEIOUaeiou", *p)) {
            *q++ = *p;
        }
        p++;
    }
    *q = '\0';
}

int main(int argc, char* argv[]) {
    if (argc < 2) {
        HandleError("Usage: <program> <output_file>");
    }

    int fd = open(argv[1], O_WRONLY | O_CREAT | O_TRUNC, 0644);
    if (fd == -1) {
        HandleError("Cannot open file");
    }

    char buffer[1024];
    ssize_t bytesRead;

    while ((bytesRead = read(STDIN_FILENO, buffer, sizeof(buffer) - 1)) > 0) {
        buffer[bytesRead] = '\0';
        RemoveVowels(buffer);
        write(fd, buffer, strlen(buffer));
        write(fd, "\n", 1);
    }

    close(fd);

    return 0;
}
```

Протокол работы программы

Тестирование:

```
artemdelgray@artemdelgray-VirtualBox:~/Загрузки/Telegram Desktop$ gcc -o p parent.c
artemdelgray@artemdelgray-VirtualBox:~/Загрузки/Telegram Desktop$ gcc -o child1 child1.c
artemdelgray@artemdelgray-VirtualBox:~/Загрузки/Telegram Desktop$ gcc -o child2 child2.c
artemdelgray@artemdelgray-VirtualBox:~/Загрузки/Telegram Desktop$ ./p
Введите имя файла для дочернего процесса 1: f1
Введите имя файла для дочернего процесса 2: f2
Введите строку (или 'exit' для завершения): poailoes
Введите строку (или 'exit' для завершения): potato
Введите строку (или 'exit' для завершения): hello world!
Введите строку (или 'exit' для завершения): MAI
Введите строку (или 'exit' для завершения): ////...,,,,,,
Введите строку (или 'exit' для завершения): legends
Введите строку (или 'exit' для завершения): exit
Работа завершена.
artemdelgray@artemdelgray-VirtualBox:~/Загрузки/Telegram Desktop$ S
```





Strace:

```
$ strace -f ./p
execve("./p", ["./p"], 0x7ffc4385f938 /* 46 vars */) = 0
brk(NULL)                               = 0x59ff92dc4000
arch_prctl(0x3001 /* ARCH_??? */, 0x7ffceb3dc990) = -1 EINVAL (Недопустимый аргумент)
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x71a094686000
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
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=57867, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 57867, PROT_READ, MAP_PRIVATE, 3, 0) = 0x71a094677000
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\0P\237\2\0\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 \0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0"..., 48, 848) = 48
pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0I\17\357\204\3$\f\221\2039x\324\224\323\236S"..., 68,
896) = 68
newfstatat(3, "", {st_mode=S_IFREG|0755, st_size=2220400, ...}, AT_EMPTY_PATH) = 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
mmap(NULL, 2264656, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x71a094400000
mprotect(0x71a094428000, 2023424, PROT_NONE) = 0
mmap(0x71a094428000, 1658880, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x28000) = 0x71a094428000
mmap(0x71a0945bd000, 360448, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1bd000) =
0x71a0945bd000
mmap(0x71a094616000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x215000) = 0x71a094616000
mmap(0x71a09461c000, 52816, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) =
0x71a09461c000
close(3)                                = 0
mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x71a094674000
arch_prctl(ARCH_SET_FS, 0x71a094674740) = 0
set_tid_address(0x71a094674a10)         = 5137
set_robust_list(0x71a094674a20, 24)     = 0
rseq(0x71a0946750e0, 0x20, 0, 0x53053053) = 0
mprotect(0x71a094616000, 16384, PROT_READ) = 0
mprotect(0x59ff91ea7000, 4096, PROT_READ) = 0
mprotect(0x71a0946c0000, 8192, PROT_READ) = 0
prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0
munmap(0x71a094677000, 57867)           = 0
pipe2([3, 4], 0)                        = 0
pipe2([5, 6], 0)                        = 0
write(1, "\320\222\320\262\320\265\320\264\320\270\321\202\320\265 \320\270\320\274\321\217
\321\204\320\260\320\271\320\273\320\260"..., 79Введите имя файла для дочернего процесса 1: ) =
79
getrandom("\x99\x0d\x2b\xfb\x1f\x3f\xd0\x55", 8, GRND_NONBLOCK) = 8
brk(NULL)                               = 0x59ff92dc4000
brk(0x59ff92de5000)                     = 0x59ff92de5000
read(0,
"\n", 1)                        = 1
write(1, "\320\222\320\262\320\265\320\264\320\270\321\202\320\265 \320\270\320\274\321\217
\321\204\320\260\320\271\320\273\320\260"..., 79Введите имя файла для дочернего процесса 2: ) =
79
read(0,
"\n", 1)                        = 1
clone(child_stack=NULL, flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLDstrace: Process
5138 attached
, child_tidptr=0x71a094674a10) = 5138
[pid  5137] clone(child_stack=NULL, flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD
<unfinished ...>
[pid  5138] set_robust_list(0x71a094674a20, 24strace: Process 5139 attached
) = 0
[pid  5139] set_robust_list(0x71a094674a20, 24 <unfinished ...>
[pid  5138] close(4 <unfinished ...>
[pid  5139] <... set_robust_list resumed>) = 0
[pid  5139] close(6 <unfinished ...>
[pid  5138] <... close resumed>)        = 0
[pid  5138] close(5)                    = 0
```

```
[pid  5138] close(6 <unfinished ...>
[pid  5139] <... close resumed>)        = 0
[pid  5138] <... close resumed>)        = 0
[pid  5139] close(3 <unfinished ...>
[pid  5138] dup2(3, 0 <unfinished ...>
[pid  5139] <... close resumed>)        = 0
[pid  5138] <... dup2 resumed>)         = 0
[pid  5139] close(4 <unfinished ...>
[pid  5138] close(3 <unfinished ...>
[pid  5139] <... close resumed>)        = 0
[pid  5138] <... close resumed>)        = 0
[pid  5138] execve("./child1", ["./child1", ""], 0x7ffceb3dcb68 /* 46 vars */ <unfinished ...>
[pid  5137] <... clone resumed>, child_tidptr=0x71a094674a10) = 5139
[pid  5139] dup2(5, 0 <unfinished ...>
[pid  5137] close(3 <unfinished ...>
[pid  5139] <... dup2 resumed>)         = 0
[pid  5139] close(5)                    = 0
[pid  5139] execve("./child2", ["./child2", ""], 0x7ffceb3dcb68 /* 46 vars */ <unfinished ...>
[pid  5137] <... close resumed>)        = 0
[pid  5137] close(5)                    = 0
[pid  5137] write(1, "\320\222\320\262\320\265\320\264\320\270\321\202\320\265
\321\201\321\202\321\200\320\276\320\272\321\203 (\320\270\320"..., 73Введите строку (или 'exit'
для завершения): ) = 73
[pid  5137] read(0,  <unfinished ...>
[pid  5138] <... execve resumed>)       = 0
[pid  5138] brk(NULL <unfinished ...>
[pid  5139] <... execve resumed>)       = 0
[pid  5139] brk(NULL <unfinished ...>
[pid  5138] <... brk resumed>)          = 0x62c288e2a000
[pid  5139] <... brk resumed>)          = 0x601626c6f000
[pid  5139] arch_prctl(0x3001 /* ARCH_??? */, 0x7ffcd24c4d10 <unfinished ...>
[pid  5138] arch_prctl(0x3001 /* ARCH_??? */, 0x7fff10a5fca0) = -1 EINVAL (Недопустимый
аргумент)
[pid  5138] mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
0x71b068172000
[pid  5138] access("/etc/ld.so.preload", R_OK) = -1 ENOENT (Нет такого файла или каталога)
[pid  5138] openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
[pid  5138] newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=57867, ...}, AT_EMPTY_PATH) = 0
[pid  5138] mmap(NULL, 57867, PROT_READ, MAP_PRIVATE, 3, 0) = 0x71b068163000
[pid  5138] close(3)                    = 0
[pid  5138] openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
[pid  5138] read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0P\237\2\0\0\0\0"..., 832) =
832
[pid  5138] 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
[pid  5138] pread64(3, "\4\0\0\0 \0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0"..., 48,
848) = 48
[pid  5138] pread64(3,
"\4\0\0\0\24\0\0\0\3\0\0\0GNU\0I\17\357\204\3$\f\221\2039x\324\224\323\236S"..., 68, 896) = 68
[pid  5138] newfstatat(3, "", {st_mode=S_IFREG|0755, st_size=2220400, ...}, AT_EMPTY_PATH) = 0
[pid  5138] 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
[pid  5138] mmap(NULL, 2264656, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x71b067e00000
[pid  5138] mprotect(0x71b067e28000, 2023424, PROT_NONE) = 0
[pid  5138] mmap(0x71b067e28000, 1658880, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000) = 0x71b067e28000
[pid  5139] <... arch_prctl resumed>)   = -1 EINVAL (Недопустимый аргумент)
[pid  5139] mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0 <unfinished
...>
[pid  5138] mmap(0x71b067fbd000, 360448, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x1bd000) = 0x71b067fbd000
[pid  5139] <... mmap resumed>)         = 0x74cc1e9df000
[pid  5138] mmap(0x71b068016000, 24576, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x215000 <unfinished ...>
[pid  5139] access("/etc/ld.so.preload", R_OK <unfinished ...>
[pid  5138] <... mmap resumed>)         = 0x71b068016000
[pid  5139] <... access resumed>)       = -1 ENOENT (Нет такого файла или каталога)
[pid  5138] mmap(0x71b06801c000, 52816, PROT_READ|PROT_WRITE,
```

```
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0 <unfinished ...>
[pid  5139] openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC <unfinished ...>
[pid  5138] <... mmap resumed>)         = 0x71b06801c000
[pid  5139] <... openat resumed>)       = 3
[pid  5138] close(3 <unfinished ...>
[pid  5139] newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=57867, ...}, AT_EMPTY_PATH) = 0
[pid  5139] mmap(NULL, 57867, PROT_READ, MAP_PRIVATE, 3, 0 <unfinished ...>
[pid  5138] <... close resumed>)        = 0
[pid  5139] <... mmap resumed>)         = 0x74cc1e9d0000
[pid  5138] mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0 <unfinished
...>
[pid  5139] close(3)                    = 0
[pid  5139] openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC <unfinished
...>
[pid  5138] <... mmap resumed>)         = 0x71b068160000
[pid  5139] <... openat resumed>)       = 3
[pid  5138] arch_prctl(ARCH_SET_FS, 0x71b068160740 <unfinished ...>
[pid  5139] read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0P\237\2\0\0\0\0"..., 832) =
832
[pid  5138] <... arch_prctl resumed>)   = 0
[pid  5139] pread64(3,  <unfinished ...>
[pid  5138] set_tid_address(0x71b068160a10 <unfinished ...>
[pid  5139] <... pread64
resumed>"\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
[pid  5138] <... set_tid_address resumed>) = 5138
[pid  5138] set_robust_list(0x71b068160a20, 24 <unfinished ...>
[pid  5139] pread64(3,  <unfinished ...>
[pid  5138] <... set_robust_list resumed>) = 0
[pid  5139] <... pread64 resumed>"\4\0\0\0
\0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0"..., 48, 848) = 48
[pid  5138] rseq(0x71b0681610e0, 0x20, 0, 0x53053053 <unfinished ...>
[pid  5139] pread64(3,  <unfinished ...>
[pid  5138] <... rseq resumed>)         = 0
[pid  5139] <... pread64
resumed>"\4\0\0\0\24\0\0\0\3\0\0\0GNU\0I\17\357\204\3$\f\221\2039x\324\224\323\236S"..., 68,
896) = 68
[pid  5138] mprotect(0x71b068016000, 16384, PROT_READ <unfinished ...>
[pid  5139] newfstatat(3, "", {st_mode=S_IFREG|0755, st_size=2220400, ...}, AT_EMPTY_PATH) = 0
[pid  5139] 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
[pid  5139] mmap(NULL, 2264656, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x74cc1e600000
[pid  5139] mprotect(0x74cc1e628000, 2023424, PROT_NONE) = 0
[pid  5139] mmap(0x74cc1e628000, 1658880, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000) = 0x74cc1e628000
[pid  5138] <... mprotect resumed>)     = 0
[pid  5139] mmap(0x74cc1e7bd000, 360448, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x1bd000 <unfinished ...>
[pid  5138] mprotect(0x62c287e72000, 4096, PROT_READ <unfinished ...>
[pid  5139] <... mmap resumed>)         = 0x74cc1e7bd000
[pid  5138] <... mprotect resumed>)     = 0
[pid  5139] mmap(0x74cc1e816000, 24576, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x215000) = 0x74cc1e816000
[pid  5138] mprotect(0x71b0681ac000, 8192, PROT_READ <unfinished ...>
[pid  5139] mmap(0x74cc1e81c000, 52816, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0 <unfinished ...>
[pid  5138] <... mprotect resumed>)     = 0
[pid  5139] <... mmap resumed>)         = 0x74cc1e81c000
[pid  5138] prlimit64(0, RLIMIT_STACK, NULL,  <unfinished ...>
[pid  5139] close(3 <unfinished ...>
[pid  5138] <... prlimit64 resumed>{rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0
[pid  5139] <... close resumed>)        = 0
[pid  5139] mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0 <unfinished
...>
[pid  5138] munmap(0x71b068163000, 57867 <unfinished ...>
[pid  5139] <... mmap resumed>)         = 0x74cc1e9cd000
[pid  5138] <... munmap resumed>)       = 0
[pid  5139] arch_prctl(ARCH_SET_FS, 0x74cc1e9cd740 <unfinished ...>
[pid  5138] openat(AT_FDCWD, "", O_WRONLY|O_CREAT|O_TRUNC, 0644 <unfinished ...>
```

```
[pid  5139] <... arch_prctl resumed>)   = 0
[pid  5139] set_tid_address(0x74cc1e9cda10) = 5139
[pid  5139] set_robust_list(0x74cc1e9cda20, 24 <unfinished ...>
[pid  5138] <... openat resumed>)       = -1 ENOENT (Нет такого файла или каталога)
[pid  5139] <... set_robust_list resumed>) = 0
[pid  5139] rseq(0x74cc1e9ce0e0, 0x20, 0, 0x53053053 <unfinished ...>
[pid  5138] write(2, "Cannot open file", 16 <unfinished ...>
Cannot open file[pid  5139] <... rseq resumed>)         = 0
[pid  5138] <... write resumed>)        = 16
[pid  5139] mprotect(0x74cc1e816000, 16384, PROT_READ <unfinished ...>
[pid  5138] write(2, "\n", 1
)           = 1
[pid  5138] exit_group(1 <unfinished ...>
[pid  5139] <... mprotect resumed>)     = 0
[pid  5138] <... exit_group resumed>)   = ?
[pid  5139] mprotect(0x601625add000, 4096, PROT_READ <unfinished ...>
[pid  5138] +++ exited with 1 +++
[pid  5137] <... read resumed>0x7ffceb3dc9af, 1) = ? ERESTARTSYS (To be restarted if SA_RESTART
is set)
[pid  5137] --- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=5138, si_uid=1000,
si_status=1, si_utime=0, si_stime=0} ---
[pid  5139] <... mprotect resumed>)     = 0
[pid  5137] read(0,  <unfinished ...>
[pid  5139] mprotect(0x74cc1ea19000, 8192, PROT_READ) = 0
[pid  5139] prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0
[pid  5139] munmap(0x74cc1e9d0000, 57867) = 0
[pid  5139] openat(AT_FDCWD, "", O_WRONLY|O_CREAT|O_TRUNC, 0644) = -1 ENOENT (Нет такого файла
или каталога)
[pid  5139] write(2, "Cannot open file", 16Cannot open file) = 16
[pid  5139] write(2, "\n", 1
)           = 1
[pid  5139] exit_group(1)               = ?
[pid  5139] +++ exited with 1 +++
<... read resumed>0x7ffceb3dc9af, 1)    = ? ERESTARTSYS (To be restarted if SA_RESTART is set)
--- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=5139, si_uid=1000, si_status=1,
si_utime=0, si_stime=0} ---
read(0,
"\n", 1)                        = 1
write(4, "\0", 1)                       = -1 EPIPE (Обрыв канала)
--- SIGPIPE {si_signo=SIGPIPE, si_code=SI_USER, si_pid=5137, si_uid=1000} ---
+++ killed by SIGPIPE +++
```

Вывод

С помощью системных вызовов и возможностей языка Си можно удобно организовать взаимодействие между разными процессами, связать их, передавать и обрабатывать данные между ними. Основными системными вызовами для этого являются: fork(), pipe(), а также read(), write() и open().