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-215Б-23

Студент: Лапенко К.А.

Преподаватель: Миронов Е.С.

Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата: 10.02.24

Москва, 2024

**Постановка задачи**

**Вариант 14.**

Родительский процесс создает два дочерних процесса. Перенаправление стандартных потоков ввода-вывода показано на картинке выше. Child1 и Child2 можно «соединить» между собой дополнительным каналом. Родительский и дочерний процесс должны быть представлены разными программами. Родительский процесс принимает от пользователя строки произвольной длины и пересылает их в pipe1. Процесс child1 и child2 производят работу над строками. Child2 пересылает результат своей работы родительскому процессу. Родительский процесс полученный результат выводит в стандартный поток вывода.

14 вариант) Child1 переводит строки в нижний регистр. Child2 убирает все задвоенные пробелы.

**Общий метод и алгоритм решения**

Использованные системные вызовы:

* pid\_t fork(void); – создает дочерний процесс.
* int pipe(int \*fd); - создает канал для взаимодействия разных процессов
* int dup2(int fd, int fd2) ; - дублирует fd на fd2, закрыв fd2 и открыв его в том же файле.
* int execl(const char\* path, const char\* arg, ...); - заменяет текущий процесс новым.
* int close(int fd) ; - закрывает переданный fd
* wait(NULL); - ждет завершения процесса
* read(int fd, void \*buf, size\_t count); - читает данные из файлового дескриптора fd в буфер
* write(int fd, const void \*buf, size\_t count); - записывает данные из буфера в файловый дескриптор fd.

Алгоритм решения:

* Родительский процесс (parent.c)
* Создает два канала: pipe1 и pipe2.
* Создает первый дочерний процесс (child1) с помощью fork().
* Создает второй дочерний процесс (child2) с помощью fork().
* Закрывает ненужные концы каналов.
* Читает строки от пользователя и записывает их в pipe1.
* Закрывает конец записи pipe1 после завершения ввода.
* Создает первый дочерний процесс:
  + Закрывает ненужные концы каналов.
  + Перенаправляет стандартный ввод (stdin) на чтение из pipe1 с помощью dup2(pipe1[0], STDIN\_FILENO).
  + Перенаправляет стандартный вывод (stdout) на запись в pipe2 с помощью dup2(pipe2[1], STDOUT\_FILENO).
  + Закрывает оставшиеся концы каналов.
  + Запускает программу child1, которая читает данные из pipe1, переводит их в нижний регистр и записывает результат в pipe2.
* Создает второй дочерний процесс:
  + Закрывает ненужные концы каналов.
  + Перенаправляет стандартный ввод (stdin) на чтение из pipe2 с помощью dup2(pipe2[0], STDIN\_FILENO).
  + Закрывает оставшиеся концы каналов.
  + Запускает программу child2, которая читает данные из pipe2, убирает лишние пробелы и записывает результат обратно в pipe2.
* Ожидает завершения дочерних процессов с помощью wait().
* Читает окончательный результат из pipe2 и выводит его на экран.
* Закрывает конец чтения pipe2.

**Дочерний процесс child1 (child1.c)**

* Программа входит в бесконечный цикл, пока не достигнет конца ввода (EOF).
* В каждой итерации цикла: Считывает один символ с помощью getchar().
* Преобразует символ в нижний регистр с помощью функции tolower().
* Выводит преобразованный символ с помощью putchar().
* Когда ввод заканчивается, программа завершает работу.

**Дочерний процесс child2 (child2.c)**

* Программа входит в бесконечный цикл, пока не достигнет конца ввода (EOF).
* В каждой итерации цикла: Считывает один символ с помощью getchar().
* Если символ является пробелом или табуляцией: Проверяет, был ли предыдущий символ также пробелом (с помощью переменной last\_was\_space).
* Если предыдущий символ не был пробелом, выводит один пробел.
* Устанавливает last\_was\_space = true, чтобы отслеживать, что текущий символ — пробел.
* Если символ не является пробелом или табуляцией: Выводит символ как есть.
* Устанавливает last\_was\_space = false, чтобы сбросить флаг пробела.
* Когда ввод заканчивается (например, пользователь нажимает Ctrl+D), программа завершает работу.

**Код программы**

**parent.c**

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <string.h>

#include <sys/wait.h>

#define MAX\_LINE 1000

int main() {

int pipe1[2], pipe2[2];

pid\_t child1, child2;

// Создаем два канала

if (pipe(pipe1) == -1 || pipe(pipe2) == -1) {

perror("Ошибка создания канала");

exit(1);

}

// Создаем первый дочерний процесс

if ((child1 = fork()) == -1) {

perror("Ошибка создания первого дочернего процесса");

exit(1);

}

if (child1 == 0) {

// Код для первого дочернего процесса (child1)

close(pipe1[1]); // Закрываем конец записи pipe1

close(pipe2[0]); // Закрываем конец чтения pipe2

dup2(pipe1[0], STDIN\_FILENO); // Перенаправляем чтение с pipe1 на stdin

dup2(pipe2[1], STDOUT\_FILENO); // Перенаправляем вывод на pipe2

close(pipe1[0]);

close(pipe2[1]);

execl("./child1", "child1", NULL);

perror("Ошибка execl для child1");

exit(1);

}

// Создаем второй дочерний процесс

if ((child2 = fork()) == -1) {

perror("Ошибка создания второго дочернего процесса");

exit(1);

}

if (child2 == 0) {

// Код для второго дочернего процесса (child2)

close(pipe1[0]);

close(pipe1[1]); // Не используем pipe1

close(pipe2[1]); // Закрываем конец записи pipe2

dup2(pipe2[0], STDIN\_FILENO); // Перенаправляем pipe2 на stdin

close(pipe2[0]);

execl("./child2", "child2", NULL);

perror("Ошибка execl для child2");

exit(1);

}

// Родительский процесс

close(pipe1[0]); // Закрываем конец чтения pipe1

close(pipe2[0]); // Закрываем конец чтения pipe2

// close(pipe2[1]); // Не закрываем конец записи pipe2 для родителя

char line[MAX\_LINE];

printf("Введите строки (Ctrl+D для завершения):\n");

// Чтение строк от пользователя и пересылка их в child1

while (fgets(line, MAX\_LINE, stdin) != NULL) {

write(pipe1[1], line, strlen(line)); //записывает данные из line в канал pipe1

}

close(pipe1[1]); // Закрываем конец записи pipe1 после ввода

// Ожидание завершения дочерних процессов

wait(NULL); // Ждем завершения child1

wait(NULL); // Ждем завершения child2

// Чтение преобразованного текста из pipe2

char buffer[MAX\_LINE];

ssize\_t bytesRead;

printf("Преобразованный текст:\n");

while ((bytesRead = read(pipe2[0], buffer, MAX\_LINE)) > 0) {

write(STDOUT\_FILENO, buffer, bytesRead);

}

close(pipe2[0]); // Закрываем конец чтения pipe2

printf("\nВсе процессы завершены.\n");

return 0;

}

**child1.c**

#include <stdio.h>

#include <ctype.h>

int main() {

char buffer[1024];

int c;

while ((c = getchar()) != EOF) {

putchar(tolower(c));

}

return 0;

}

**child2.c**

#include <stdio.h>

#include <stdbool.h>

int main() {

char buffer[1024];

int c;

bool last\_was\_space = false;

while ((c = getchar()) != EOF) {

if (c == ' ' || c == '\t') {

if (!last\_was\_space) {

putchar(' ');

last\_was\_space = true;

}

} else {

putchar(c);

last\_was\_space = false;

}

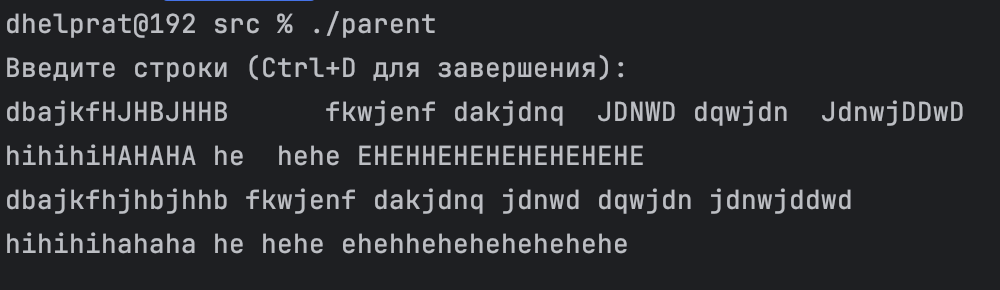
}

return 0;

}

**Протокол работы программы**

**Тестирование:**



**Strace:**

$ strace -f ./parent

execve("./parent", ["./parent"], 0x7fffe4680218 /\* 32 vars \*/) = 0

brk(NULL) = 0x564a66f35000

arch\_prctl(0x3001 /\* ARCH\_??? \*/, 0x7fff31d158d0) = -1 EINVAL (Invalid argument)

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f5e02b4c000

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

newfstatat(3, "", {st\_mode=S\_IFREG|0644, st\_size=22315, ...}, AT\_EMPTY\_PATH) = 0

mmap(NULL, 22315, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7f5e02b46000

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) = 0x7f5e0291d000

mprotect(0x7f5e02945000, 2023424, PROT\_NONE) = 0

mmap(0x7f5e02945000, 1658880, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x28000) = 0x7f5e02945000

mmap(0x7f5e02ada000, 360448, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1bd000) = 0x7f5e02ada000

mmap(0x7f5e02b33000, 24576, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x215000) = 0x7f5e02b33000

mmap(0x7f5e02b39000, 52816, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7f5e02b39000

close(3) = 0

mmap(NULL, 12288, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f5e0291a000

arch\_prctl(ARCH\_SET\_FS, 0x7f5e0291a740) = 0

set\_tid\_address(0x7f5e0291aa10) = 5658

set\_robust\_list(0x7f5e0291aa20, 24) = 0

rseq(0x7f5e0291b0e0, 0x20, 0, 0x53053053) = 0

mprotect(0x7f5e02b33000, 16384, PROT\_READ) = 0

mprotect(0x564a5a9ce000, 4096, PROT\_READ) = 0

mprotect(0x7f5e02b86000, 8192, PROT\_READ) = 0

prlimit64(0, RLIMIT\_STACK, NULL, {rlim\_cur=8192\*1024, rlim\_max=RLIM64\_INFINITY}) = 0

munmap(0x7f5e02b46000, 22315) = 0

pipe2([3, 4], 0) = 0

pipe2([5, 6], 0) = 0

clone(child\_stack=NULL, flags=CLONE\_CHILD\_CLEARTID|CLONE\_CHILD\_SETTID|SIGCHLDstrace: Process 5659 attached

, child\_tidptr=0x7f5e0291aa10) = 5659

[pid 5659] set\_robust\_list(0x7f5e0291aa20, 24 <unfinished ...>

[pid 5658] clone(child\_stack=NULL, flags=CLONE\_CHILD\_CLEARTID|CLONE\_CHILD\_SETTID|SIGCHLD <unfinished ...>

[pid 5659] <... set\_robust\_list resumed>) = 0

[pid 5659] close(4) = 0

strace: Process 5660 attached

[pid 5658] <... clone resumed>, child\_tidptr=0x7f5e0291aa10) = 5660

[pid 5659] close(5 <unfinished ...>

[pid 5658] close(3 <unfinished ...>

[pid 5660] set\_robust\_list(0x7f5e0291aa20, 24 <unfinished ...>

[pid 5658] <... close resumed>) = 0

[pid 5659] <... close resumed>) = 0

[pid 5658] close(5 <unfinished ...>

[pid 5660] <... set\_robust\_list resumed>) = 0

[pid 5658] <... close resumed>) = 0

[pid 5659] dup2(3, 0 <unfinished ...>

[pid 5658] newfstatat(1, "", <unfinished ...>

[pid 5660] close(3 <unfinished ...>

[pid 5658] <... newfstatat resumed>{st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0x2), ...}, AT\_EMPTY\_PATH) = 0

[pid 5659] <... dup2 resumed>) = 0

[pid 5658] getrandom( <unfinished ...>

[pid 5660] <... close resumed>) = 0

[pid 5658] <... getrandom resumed>"\x78\x99\xfa\x5c\x45\x7b\xf8\x30", 8, GRND\_NONBLOCK) = 8

[pid 5659] dup2(6, 1 <unfinished ...>

[pid 5658] brk(NULL <unfinished ...>

[pid 5660] close(4 <unfinished ...>

[pid 5658] <... brk resumed>) = 0x564a66f35000

[pid 5659] <... dup2 resumed>) = 1

[pid 5658] brk(0x564a66f56000 <unfinished ...>

[pid 5660] <... close resumed>) = 0

[pid 5658] <... brk resumed>) = 0x564a66f56000

[pid 5659] close(3 <unfinished ...>

[pid 5658] 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\320\270 (Ctr"..., 66 <unfinished ...>

Введите строки (Ctrl+D для завершения):

[pid 5660] close(6 <unfinished ...>

[pid 5658] <... write resumed>) = 66

[pid 5659] <... close resumed>) = 0

[pid 5658] newfstatat(0, "", <unfinished ...>

[pid 5660] <... close resumed>) = 0

[pid 5658] <... newfstatat resumed>{st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0x2), ...}, AT\_EMPTY\_PATH) = 0

[pid 5659] close(6 <unfinished ...>

[pid 5658] read(0, <unfinished ...>

[pid 5660] dup2(5, 0 <unfinished ...>

[pid 5659] <... close resumed>) = 0

[pid 5660] <... dup2 resumed>) = 0

[pid 5659] execve("./child1", ["child1"], 0x7fff31d15aa8 /\* 32 vars \*/ <unfinished ...>

[pid 5660] close(5) = 0

[pid 5660] execve("./child2", ["child2"], 0x7fff31d15aa8 /\* 32 vars \*/) = 0

[pid 5659] <... execve resumed>) = 0

[pid 5660] brk(NULL <unfinished ...>

[pid 5659] brk(NULL <unfinished ...>

[pid 5660] <... brk resumed>) = 0x559b100dd000

[pid 5659] <... brk resumed>) = 0x564cafd3d000

[pid 5660] arch\_prctl(0x3001 /\* ARCH\_??? \*/, 0x7fff8a7d8910 <unfinished ...>

[pid 5659] arch\_prctl(0x3001 /\* ARCH\_??? \*/, 0x7ffdd12b42f0 <unfinished ...>

[pid 5660] <... arch\_prctl resumed>) = -1 EINVAL (Invalid argument)

[pid 5659] <... arch\_prctl resumed>) = -1 EINVAL (Invalid argument)

[pid 5660] mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0 <unfinished ...>

[pid 5659] mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0 <unfinished ...>

[pid 5660] <... mmap resumed>) = 0x7fb5ce813000

[pid 5659] <... mmap resumed>) = 0x7f291c4b6000

[pid 5660] access("/etc/ld.so.preload", R\_OK <unfinished ...>

[pid 5659] access("/etc/ld.so.preload", R\_OK <unfinished ...>

[pid 5660] <... access resumed>) = -1 ENOENT (No such file or directory)

[pid 5659] <... access resumed>) = -1 ENOENT (No such file or directory)

[pid 5660] openat(AT\_FDCWD, "/etc/ld.so.cache", O\_RDONLY|O\_CLOEXEC <unfinished ...>

[pid 5659] openat(AT\_FDCWD, "/etc/ld.so.cache", O\_RDONLY|O\_CLOEXEC <unfinished ...>

[pid 5660] <... openat resumed>) = 3

[pid 5659] <... openat resumed>) = 3

[pid 5660] newfstatat(3, "", <unfinished ...>

[pid 5659] newfstatat(3, "", <unfinished ...>

[pid 5660] <... newfstatat resumed>{st\_mode=S\_IFREG|0644, st\_size=22315, ...}, AT\_EMPTY\_PATH) = 0

[pid 5659] <... newfstatat resumed>{st\_mode=S\_IFREG|0644, st\_size=22315, ...}, AT\_EMPTY\_PATH) = 0

[pid 5660] mmap(NULL, 22315, PROT\_READ, MAP\_PRIVATE, 3, 0 <unfinished ...>

[pid 5659] mmap(NULL, 22315, PROT\_READ, MAP\_PRIVATE, 3, 0 <unfinished ...>

[pid 5660] <... mmap resumed>) = 0x7fb5ce80d000

[pid 5659] <... mmap resumed>) = 0x7f291c4b0000

[pid 5660] close(3 <unfinished ...>

[pid 5659] close(3 <unfinished ...>

[pid 5660] <... close resumed>) = 0

[pid 5659] <... close resumed>) = 0

[pid 5660] openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/libc.so.6", O\_RDONLY|O\_CLOEXEC <unfinished ...>

[pid 5659] openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/libc.so.6", O\_RDONLY|O\_CLOEXEC <unfinished ...>

[pid 5660] <... openat resumed>) = 3

[pid 5659] <... openat resumed>) = 3

[pid 5660] read(3, <unfinished ...>

[pid 5659] read(3, <unfinished ...>

[pid 5660] <... read resumed>"\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

[pid 5659] <... read resumed>"\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

[pid 5660] pread64(3, <unfinished ...>

[pid 5659] pread64(3, <unfinished ...>

[pid 5660] <... 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 5659] <... 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 5660] pread64(3, <unfinished ...>

[pid 5659] pread64(3, <unfinished ...>

[pid 5660] <... 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 5659] <... 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 5660] pread64(3, <unfinished ...>

[pid 5659] pread64(3, <unfinished ...>

[pid 5660] <... 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 5659] <... 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 5660] newfstatat(3, "", <unfinished ...>

[pid 5659] newfstatat(3, "", <unfinished ...>

[pid 5660] <... newfstatat resumed>{st\_mode=S\_IFREG|0755, st\_size=2220400, ...}, AT\_EMPTY\_PATH) = 0

[pid 5659] <... newfstatat resumed>{st\_mode=S\_IFREG|0755, st\_size=2220400, ...}, AT\_EMPTY\_PATH) = 0

[pid 5660] pread64(3, <unfinished ...>

[pid 5659] pread64(3, <unfinished ...>

[pid 5660] <... 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 5659] <... 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 5660] mmap(NULL, 2264656, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0 <unfinished ...>

[pid 5659] mmap(NULL, 2264656, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0 <unfinished ...>

[pid 5660] <... mmap resumed>) = 0x7fb5ce5e4000

[pid 5659] <... mmap resumed>) = 0x7f291c287000

[pid 5660] mprotect(0x7fb5ce60c000, 2023424, PROT\_NONE <unfinished ...>

[pid 5659] mprotect(0x7f291c2af000, 2023424, PROT\_NONE <unfinished ...>

[pid 5660] <... mprotect resumed>) = 0

[pid 5659] <... mprotect resumed>) = 0

[pid 5660] mmap(0x7fb5ce60c000, 1658880, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x28000 <unfinished ...>

[pid 5659] mmap(0x7f291c2af000, 1658880, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x28000 <unfinished ...>

[pid 5660] <... mmap resumed>) = 0x7fb5ce60c000

[pid 5659] <... mmap resumed>) = 0x7f291c2af000

[pid 5660] mmap(0x7fb5ce7a1000, 360448, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1bd000 <unfinished ...>

[pid 5659] mmap(0x7f291c444000, 360448, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1bd000 <unfinished ...>

[pid 5660] <... mmap resumed>) = 0x7fb5ce7a1000

[pid 5659] <... mmap resumed>) = 0x7f291c444000

[pid 5660] mmap(0x7fb5ce7fa000, 24576, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x215000 <unfinished ...>

[pid 5659] mmap(0x7f291c49d000, 24576, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x215000 <unfinished ...>

[pid 5660] <... mmap resumed>) = 0x7fb5ce7fa000

[pid 5659] <... mmap resumed>) = 0x7f291c49d000

[pid 5660] mmap(0x7fb5ce800000, 52816, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7fb5ce800000

[pid 5659] mmap(0x7f291c4a3000, 52816, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0 <unfinished ...>

[pid 5660] close(3 <unfinished ...>

[pid 5659] <... mmap resumed>) = 0x7f291c4a3000

[pid 5660] <... close resumed>) = 0

[pid 5659] close(3 <unfinished ...>

[pid 5660] mmap(NULL, 12288, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0 <unfinished ...>

[pid 5659] <... close resumed>) = 0

[pid 5660] <... mmap resumed>) = 0x7fb5ce5e1000

[pid 5659] mmap(NULL, 12288, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0 <unfinished ...>

[pid 5660] arch\_prctl(ARCH\_SET\_FS, 0x7fb5ce5e1740 <unfinished ...>

[pid 5659] <... mmap resumed>) = 0x7f291c284000

[pid 5660] <... arch\_prctl resumed>) = 0

[pid 5659] arch\_prctl(ARCH\_SET\_FS, 0x7f291c284740 <unfinished ...>

[pid 5660] set\_tid\_address(0x7fb5ce5e1a10 <unfinished ...>

[pid 5659] <... arch\_prctl resumed>) = 0

[pid 5660] <... set\_tid\_address resumed>) = 5660

[pid 5659] set\_tid\_address(0x7f291c284a10 <unfinished ...>

[pid 5660] set\_robust\_list(0x7fb5ce5e1a20, 24 <unfinished ...>

[pid 5659] <... set\_tid\_address resumed>) = 5659

[pid 5660] <... set\_robust\_list resumed>) = 0

[pid 5659] set\_robust\_list(0x7f291c284a20, 24 <unfinished ...>

[pid 5660] rseq(0x7fb5ce5e20e0, 0x20, 0, 0x53053053 <unfinished ...>

[pid 5659] <... set\_robust\_list resumed>) = 0

[pid 5660] <... rseq resumed>) = 0

[pid 5659] rseq(0x7f291c2850e0, 0x20, 0, 0x53053053) = 0

[pid 5660] mprotect(0x7fb5ce7fa000, 16384, PROT\_READ) = 0

[pid 5659] mprotect(0x7f291c49d000, 16384, PROT\_READ <unfinished ...>

[pid 5660] mprotect(0x559ae1eaf000, 4096, PROT\_READ <unfinished ...>

[pid 5659] <... mprotect resumed>) = 0

[pid 5660] <... mprotect resumed>) = 0

[pid 5659] mprotect(0x564c87e7e000, 4096, PROT\_READ <unfinished ...>

[pid 5660] mprotect(0x7fb5ce84d000, 8192, PROT\_READ <unfinished ...>

[pid 5659] <... mprotect resumed>) = 0

[pid 5660] <... mprotect resumed>) = 0

[pid 5659] mprotect(0x7f291c4f0000, 8192, PROT\_READ <unfinished ...>

[pid 5660] prlimit64(0, RLIMIT\_STACK, NULL, <unfinished ...>

[pid 5659] <... mprotect resumed>) = 0

[pid 5660] <... prlimit64 resumed>{rlim\_cur=8192\*1024, rlim\_max=RLIM64\_INFINITY}) = 0

[pid 5659] prlimit64(0, RLIMIT\_STACK, NULL, <unfinished ...>

[pid 5660] munmap(0x7fb5ce80d000, 22315 <unfinished ...>

[pid 5659] <... prlimit64 resumed>{rlim\_cur=8192\*1024, rlim\_max=RLIM64\_INFINITY}) = 0

[pid 5660] <... munmap resumed>) = 0

[pid 5659] munmap(0x7f291c4b0000, 22315 <unfinished ...>

[pid 5660] newfstatat(0, "", <unfinished ...>

[pid 5659] <... munmap resumed>) = 0

[pid 5660] <... newfstatat resumed>{st\_mode=S\_IFIFO|0600, st\_size=0, ...}, AT\_EMPTY\_PATH) = 0

[pid 5660] getrandom( <unfinished ...>

[pid 5659] newfstatat(0, "", <unfinished ...>

[pid 5660] <... getrandom resumed>"\x95\x2d\x25\x37\xbc\x6d\xf2\x31", 8, GRND\_NONBLOCK) = 8

[pid 5659] <... newfstatat resumed>{st\_mode=S\_IFIFO|0600, st\_size=0, ...}, AT\_EMPTY\_PATH) = 0

[pid 5660] brk(NULL <unfinished ...>

[pid 5659] getrandom( <unfinished ...>

[pid 5660] <... brk resumed>) = 0x559b100dd000

[pid 5659] <... getrandom resumed>"\x86\xed\xa9\x13\x3d\xc8\x3e\xbd", 8, GRND\_NONBLOCK) = 8

[pid 5660] brk(0x559b100fe000 <unfinished ...>

[pid 5659] brk(NULL <unfinished ...>

[pid 5660] <... brk resumed>) = 0x559b100fe000

[pid 5659] <... brk resumed>) = 0x564cafd3d000

[pid 5660] read(0, <unfinished ...>

[pid 5659] brk(0x564cafd5e000) = 0x564cafd5e000

[pid 5659] read(0, <unfinished ...>

[pid 5658] <... read resumed>0x564a66f356b0, 1024) = ? ERESTARTSYS (To be restarted if SA\_RESTART is set)

[pid 5660] <... read resumed>0x559b100dd2a0, 4096) = ? ERESTARTSYS (To be restarted if SA\_RESTART is set)

[pid 5659] <... read resumed>0x564cafd3d2a0, 4096) = ? ERESTARTSYS (To be restarted if SA\_RESTART is set)

[pid 5658] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 5660] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 5658] read(0, <unfinished ...>

[pid 5659] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 5660] read(0, <unfinished ...>

[pid 5659] read(0, <unfinished ...>

[pid 5658] <... read resumed>0x564a66f356b0, 1024) = ? ERESTARTSYS (To be restarted if SA\_RESTART is set)

[pid 5660] <... read resumed>0x559b100dd2a0, 4096) = ? ERESTARTSYS (To be restarted if SA\_RESTART is set)

[pid 5659] <... read resumed>0x564cafd3d2a0, 4096) = ? ERESTARTSYS (To be restarted if SA\_RESTART is set)

[pid 5658] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 5660] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 5658] read(0, <unfinished ...>

[pid 5659] --- SIGWINCH {si\_signo=SIGWINCH, si\_code=SI\_KERNEL} ---

[pid 5660] read(0, <unfinished ...>

[pid 5659] read(0, AAA HHH FF VV

<unfinished ...>

[pid 5658] <... read resumed>"AAA HHH FF VV\n", 1024) = 17

[pid 5658] write(4, "AAA HHH FF VV\n", 17) = 17

[pid 5659] <... read resumed>"AAA HHH FF VV\n", 4096) = 17

[pid 5658] read(0, <unfinished ...>

[pid 5659] newfstatat(1, "", {st\_mode=S\_IFIFO|0600, st\_size=0, ...}, AT\_EMPTY\_PATH) = 0

[pid 5659] read(0, gg hh

<unfinished ...>

[pid 5658] <... read resumed>"gg hh \n", 1024) = 7

[pid 5658] write(4, "gg hh \n", 7) = 7

[pid 5659] <... read resumed>"gg hh \n", 4096) = 7

[pid 5658] read(0, <unfinished ...>

[pid 5659] read(0, <unfinished ...>

[pid 5658] <... read resumed>"", 1024) = 0

[pid 5658] close(4) = 0

[pid 5659] <... read resumed>"", 4096) = 0

[pid 5658] wait4(-1, <unfinished ...>

[pid 5659] write(1, "aaa hhh ff vv\ngg hh \n", 24) = 24

[pid 5660] <... read resumed>"aaa hhh ff vv\ngg hh \n", 4096) = 24

[pid 5659] exit\_group(0 <unfinished ...>

[pid 5660] newfstatat(1, "", <unfinished ...>

[pid 5659] <... exit\_group resumed>) = ?

[pid 5660] <... newfstatat resumed>{st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0x2), ...}, AT\_EMPTY\_PATH) = 0

[pid 5660] write(1, "aaa hhh ff vv\n", 14aaa hhh ff vv

) = 14

[pid 5660] write(1, "gg hh \n", 7gg hh

) = 7

[pid 5660] read(0, <unfinished ...>

[pid 5659] +++ exited with 0 +++

[pid 5658] <... wait4 resumed>NULL, 0, NULL) = 5659

[pid 5658] --- SIGCHLD {si\_signo=SIGCHLD, si\_code=CLD\_EXITED, si\_pid=5659, si\_uid=1000, si\_status=0, si\_utime=0, si\_stime=0} ---

[pid 5658] wait4(-1, ^CNULL, 0, NULL) = ? ERESTARTSYS (To be restarted if SA\_RESTART is set)

strace: Process 5658 detached

strace: Process 5660 detached

**Вывод**

Было интересно разобраться с процессами и каналами. Поняла, что важно внимательно следить за каналами и закрывать все ненужные, во избежание непредвиденных ситуаций. Самым сложным было держать в голове для какого процесса я сейчас пишу код и как он может повлиять на другие процессы, которые будут этот код выполнять.