Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Отчёт

по лабораторной работе № 4

Название: Пять системных вызовов ОС UNIX/LINUX

Дисциплина: Операционные системы

Студент ИУ7-55Б _____ Д.О. Склифасовский
(Группа) (Подпись, дата) (И.О. Фамилия)

Преподователь _____ Н.Ю. Рязанова
(Подпись, дата) (И.О. Фамилия)

Москва, 2020

Содержание

Задание 1

Написать программу, запускающую не мене двух новых процессов системным вызовом fork(). В предке вывести собственный идентификатор (функция getpid()), идентификатор группы (функция getpgrp()) и идентификаторы потомков. В процессе-потомке вывести собственный идентификатор, идентификатор предка (функция getppid()) и идентификатор группы. Убедиться, что при завершении процесса-предка потомок, который продолжает выполняться, получает идентификатор предка (PPID), равный 1 или идентификатор процесса-посредника.

Код программы:

Листинг 1 – Задание 1

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

void print_child(int child_num, char *descr)
{
    printf("child: number=%d pid=%-5d parent=%-5d group=%-5d %s\n"
        , child_num, getpid(), getppid(), getpgrp(), descr);
}

pid_t fork_child(int child_num)
{
    pid_t child = fork();
    if (child == -1)
    {
        perror("fork");
        exit(1);
    }
    else if (child == 0)
    {
```

```
20          print_child(child_num, "before sleep");
21          sleep(2);
22          print_child(child_num, "after sleep");
23          exit(0);
24      }
25      return child;
26  }
27
28  int main()
29  {
30      pid_t child_1 = fork_child(1);
31      pid_t child_2 = fork_child(2);
32
33      printf("parent: pid=%-5d, child_1=%-5d, child_2=%-5d\n",
            getpid(), child_1, child_2);
34
35      return 0;
36  }
```

Результат работы программы:

```
parent: pid=3364 , child_1=3365 , child_2=3366
child: number=1 pid=3365  parent=3364  group=3364  before sleep
child: number=2 pid=3366  parent=3364  group=3364  before sleep
mrskl1f@mrskl1f-ThinkPad-E595:~/Рабочий стол/BMSTU/Unix/Lab2$ child: number=1 pid=3365  parent=1     group=3364  after sleep
child: number=2 pid=3366  parent=1     group=3364  after sleep
```

Рисунок 1 – Результат работы программы

1 Задание 2

Написать программу по схеме первого задания, но в процессе-предке выполнить системный вызов wait(). Убедиться, что в этом случае идентификатор процесса потомка на 1 больше идентификатора процесса-предка.

Задание 3: написать программу, в которой процесс-потомок вызывает системный вызов exec(), а процесс-предок ждет завершения процесса-потомка. Следует создать не менее двух потомков.

Задание 4: написать программу, в которой предок и потомок обмениваются сообщением через программный канал. **Задание 5:** в программу с программным каналом включить собственный обработчик сигнала. Использовать сигнал для изменения хода выполнения программы.