Лабораторная работа №14

Лабораторная работа № 14. Именованные каналы.

Старовойтов Егор Сергеевич

Содержание

Цель работы

Приобретение практических навыков работы с именованными каналами.

Задание

Изучите приведённые в тексте программы server.c и client.c. Взяв данные примеры за образец, напишите аналогичные программы, внеся следующие изменения: 1. Работает не 1 клиент, а несколько (например, два). 2. Клиенты передают текущее время с некоторой периодичностью (например, раз в пять секунд). Используйте функцию sleep() для приостановки работы клиента. 3. Сервер работает не бесконечно, а прекращает работу через некоторое время (например, 30 сек). Используйте функцию clock() для определения времени работы сервера. Что будет в случае, если сервер завершит работу, не закрыв канал?

Теоретическое введение

Одним из видов взаимодействия между процессами в операционных системах является обмен сообщениями. Под сообщением понимается последовательность байтов, передаваемая от одного процесса другому. В операционных системах типа UNIX есть 3 вида межпроцессорных взаимодействий: общеюниксные (именованные каналы, сигналы), System V Interface Definition (SVID — разделяемая память, очередь сообщений, семафоры) и BSD (сокеты). Для передачи данных между неродственными процессами можно использовать механизм именованных каналов (named pipes). Данные передаются по принципу FIFO (First In First Out) (первым записан — первым прочитан), поэтому они называются также FIFO pipes или просто FIFO. Именованные каналы отличаются от неименованных наличием идентификатора канала, который представлен как специальный файл (соответственно имя именованного канала — это имя файла). Поскольку файл находится на локальной файловой системе, данное IPC используется внутри одной системы.

Ход работы

Шаг 1

Я создал необходимые файлы с помощью touch.

```
[liveuser@localhost-live ~]$ touch common.h
[liveuser@localhost-live ~]$ touch server.c
[liveuser@localhost-live ~]$ touch client.c
[liveuser@localhost-live ~]$ touch Makefile
[liveuser@localhost-live ~]$ ls
client.c  common.h  Desktop  Documents  Downloads  Makefile  Music  Pictures  Public  server.c  Templates  Videos
[liveuser@localhost-live ~]$
```

Шаг 2

Изменил исходный код файлов: - В common.h добавил заголовочные файлы time.h и unistd.h. - В client.c добавил цикл, регулирующий кол-во сообщений и команду sleep(5), для приостановки выполнения программы на 5 секунд. - В server.c добавил цикл while для контроля времени работы сервера.

```
[liveuser@localhost-live ~]$ cat common.h
#ifndef __COMMON_H__
#define __COMMON_H__

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <errno.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <time.h>
#include <unistd.h>

#define FIFO_NAME "/tmp/fifo"
#define MAX_BUFF 80

#endif /* __COMMON_H__ */
[liveuser@localhost-live ~]$
```

```
[liveuser@localhost-live ~]$ cat client.c
#include "common.h"

#define MESSAGE "Hello Server!!!\n"

int main() {
    int writefd;
    int msglen;

    printf("FIFO Client...\n");

    for (int i = 0; i < 4; i++) {
        if ((writefd = open(FIFO_NAME, O_WRONLY)) < 0) {
            fprintf(
                stderr,
                "%s: Unable to open FIFO (%s)\n",
                __FILE__,
                strerror(errno)
            );
            exit(-1);
        }

        long int current_time = time(NULL);
        char *text = ctime(&current_time);

        msglen = strlen(MESSAGE);
        if (write(writefd, MESSAGE, msglen) != msglen) {
            fprintf(
                stderr,
                "%s: Error writing to FIFO (%s)\n",
                __FILE__,
                strerror(errno)
            );
            exit(-2);
        }
        sleep(5);
    }

    close(writefd);
    exit(0);
}

[liveuser@localhost-live ~]$
```

```c
    printf("FIFO Server...\n");

    if (mknod(FIFO_NAME, S_IFIFO | 0666, 0) < 0) {
        fprintf(
            stderr,
            "%s: Unable to create the FIFO (%s)\n",
            __FILE__,
            strerror(errno)
        );
        exit(-1);
    }

    if ((readfd = open(FIFO_NAME, O_RDONLY)) < 0) {
        fprintf(
            stderr,
            "%s: Unable to open the FIFO (%s)\n",
            __FILE__,
            strerror(errno)
        );
        exit(-2);
    }

    time_t start = time(NULL);

    while (time(NULL) - start < 30) {
        while ((n = read(readfd, buff, MAX_BUFF)) > 0) {
            if (write(1, buff, n) != n) {
                fprintf(
                    stderr,
                    "%s: Output error (%s)\n",
                    __FILE__,
                    strerror(errno)
                );
                exit(-3);
            }
        }
    }

    if (unlink(FIFO_NAME) < 0) {
        fprintf(
            stderr,
            "%s: Unable to remove FIFO (%s)\n",
            __FILE__,
            strerror(errno)
        );
        exit(-4);
    }
    exit(0);
}
```

Шаг 3

Запустил программы сервера и клиента.

```
[liveuser@localhost-live ~]$ ./server
FIFO Server...
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
```

Работа сервера

```
[liveuser@localhost-live ~]$ ./client
FIFO Client...
[liveuser@localhost-live ~]$
```

Работа клиента

Если сервер завершит работу, не закрыв канал, то при повторном запуске программы сервера возникнет ошибка при создании файла канала, так как он все ещще будет существовать с прошлого раза.

```
[liveuser@localhost-live ~]$ ./server
FIFO Server...
server.c: Unable to create the FIFO (File exists)
[liveuser@localhost-live ~]$
```

Ошибка создания канала

Вывод

Я приобрел практические навыки работы с именованными каналами.

Контрольные вопросы

1. В чем ключевое отличие именованных каналов от неименованных?

Именованные каналы отличаются от неименованных наличием идентификатора канала, который представлен как специальный файл (соответственно имя именованного канала — это имя файла). Поскольку файл находится на локальной файловой системе, данное IPC используется внутри одной системы.

2. Возможно ли создание неименованного канала из командной строки?

Да, с помощью оператора |.

3. Возможно ли создание именованного канала из командной строки?

Да, с помощью команд mkfifo и mknod.

4. Опишите функцию языка C, создающую неименованный канал.

Неименованный канал создается вызовом pipe, который заносит в массив int [2] два дескриптора открытых файлов. fd[0] – открыт на чтение, fd[1] – на запись (вспомните STDIN == 0, STDOUT == 1). Канал уничтожается, когда будут закрыты все файловые дескрипторы ссылающиеся на него.

5. Опишите функцию языка C, создающую именованный канал.

например с помощью вызова unlink(2). Рассмотрим работу именованного канала на примере системы клиент–сервер. Сервер создаёт канал, читает из него текст, посылаемый клиентом, и выводит его на терминал. Вызов функции mkfifo() создаёт файл канала (с именем, заданным макросом FIFO_NAME):

6. Что будет в случае прочтения из fifo меньшего числа байтов, чем находится в канале?Большего числа байтов?

Вернется требуемое число байтов, а остаток будет ожидать следующего чтения.

7. Аналогично, что будет в случае записи в fifo меньшего числа байтов, чем позволяет буфер? Большего числа байтов?

Запись меньшего числа байтов возможна и является атомарной операцией. При записи большего числа байтов процесс блокируется до освобождения места в канале.

8. Могут ли два и более процессов читать или записывать в канал?

Да. Но одновременно можно только читать данные, запись в каналы осуществляется по очереди.

9. Опишите функцию write (тип возвращаемого значения, аргументы и логику работы).

write пишет count байтов из буфера buffer в файл handle. Запись начичнается с указателя, ассоциированного с handle. Запись происходит в режиме перезаписи или добавления, смотря как открыт файл handle. Функция write возвращает число записанных байтов.

10. Опишите функцию strerror.

Функция strerror преобразует аргумент (номер ошибки) в понятный человеку текст.