Университет ИТМО факультет программной инженерии и компьютерной техники

Отчет по лабораторной работе №1 по дисциплине «Операционные системы»

Вариант:

A=248;B=0x3FFDBBFB;C=malloc;D=125;E=195;F=nocache;G=20;H=seq;I=131;J=sum;K=futex

Работу выполнил: Щербаков Виталий Юрьевич Группа Р33122

Преподаватель: Покид Александр Владимирович

Задание:

Разработать программу на языке С, которая осуществляет следующие действия

- Создает область памяти размером А мегабайт, начинающихся с адреса В (если возможно) при помощи С=(malloc, mmap) заполненную случайными числами /dev/urandom в D потоков. Используя системные средства мониторинга определите адрес начала в адресном пространстве процесса и характеристики выделенных участков памяти. Замеры виртуальной/физической памяти необходимо снять:
- 1. До аллокации
- 2. После аллокации
- 3. После заполнения участка данными
- 4. После деаллокации
- Записывает область памяти в файлы одинакового размера Е мегабайт с использованием F=(блочного, некешируемого) обращения к диску. Размер блока ввода-вывода G байт. Преподаватель выдает в качестве задания последовательность записи/чтения блоков H=(последовательный, заданный или случайный)
- Генерацию данных и запись осуществлять в бесконечном цикле.
- В отдельных I потоках осуществлять чтение данных из файлов и подсчитывать агрегированные характеристики данных - J=(сумму, среднее значение, максимальное, минимальное значение).
- Чтение и запись данных в/из файла должна быть защищена примитивами синхронизации K=(futex, cv, sem, flock).
- По заданию преподавателя изменить приоритеты потоков и описать изменения в характеристиках программы.

Для запуска программы возможно использовать операционную систему Windows 10 или Debian/Ubuntu в виртуальном окружении.

Измерить значения затраченного процессорного времени на выполнение программы и на операции ввода-вывода используя системные утилиты.

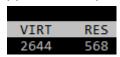
Отследить трассу системных вызовов.

Используя stap построить графики системных характеристик.

Замеры виртуальной и физической памяти:

Используем утилиту top:

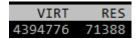
1) до аллокации



2) после аллокации



3) после заполнения данными



4) после деаллокации



Замеры затраченного времени процессором:

Используем утилиту top:



Замеры затраченного времени на операции ввода-вывода:

Используем утилиту iostat:



Адрес начала в адресном пространстве процесса и характеристики:

Получаем pid нашего процесса и используем команду pmap –x {pid}:

```
oldrim@DESKTOP-1M7HMGH:~/os lab1$ pmap -x 14666
14666:
        /mnt/c/Users/User/CLionProjects/oslab/cmake-build-wsl_profile/oslab
Address
                          RSS
                                 Dirty Mode Mapping
                 Kbytes
0000556e80a86000
                                     0 r---- oslab
                     8
0000556e80a87000
                            8
                                    0 r-x-- oslab
                           4 4
0000556e80a89000
                                    0 r---- oslab
                     4
                    4
                                   4 r---- oslab
4 rw--- oslab
0000556e80a8a000
0000556e80a8b000
                                  80 rw--- [ anon
32 rw--- [ anon
                          80
32
0000556e81695000
                  132
00007f71a0000000
                   132
00007f71a0021000 65404
                            0
                                    0 ----
                                               [ anon
                  132
                             32
                                   32 rw---
                                               [ anon
00007f71a8000000
                                               anon anon
00007f71a8021000
                  65404
                            0
                                    0 ----
00007f71ac000000
                    132
                             32
                                    32 rw---
00007f71ac021000
                  65404
                             Θ
                                     а
                                                 anon
```

Трасса системных вызовов при помощи утилиты strace:

Используем команду strace ./main:

Вывод:

В ходе данной лабораторной работы я обучился основам работы с языком системного программирования С и получил новые знания об устройстве Unix операционных систем. Также, я научился пользоваться многочисленными утилитами мониторинга для получения информации о системных характеристиках.