ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ»

Факультет безопасности информационных технологий

Дисциплина:
“Операционные системы”

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2

Выполнил:
Студент гр. N3249
Шарифуллин Ильдан Айдарович

Проверил:
Савков С.В.

Санкт-Петербург
2022г.

Задание:

1. Написать программу выделения памяти и заполнения ее нулями с шагом, равным размеру страницы памяти (mmap, VirtualAlloc)
2. Составить график свободной памяти
3. Ознакомиться с работой демона OOM Killer в Linux
4. Достичь сообщения о невозможности выделить память в Windows

Ход работы:

1. **Написать программу выделения памяти и заполнения ее нулями с шагом, равным размеру страницы памяти (mmap, VirtualAlloc)**

Membomb для Linux:

```
#include <unistd.h>
#include <stdlib.h>
#include <sys/mman.h>

int main() {
  long pgSize = sysconf(_SC_PAGESIZE);
  while(1) {
    char* block = mmap(0, pgSize, PROT_READ|PROT_WRITE, MAP_PRIVATE |
MAP_ANONYMOUS, -1, 0);
    if (block) {
    for (long i = 0; i < pgSize; i++)
      block[i] = 0;
    }
    usleep(1);
  }
  return 0;
}
```

Запись логов на баше для Linux:

```
#!/bin/bash
free -s 0.1 -b >> log
```

Membomb для Windows:

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

```
int main() {
  long pgSize = 4096;
  while(1) {
    char* block = (char*)VirtualAlloc(0, pgSize, MEM_COMMIT, PAGE_READWRITE);
    if (block) {
    for (long i = 0; i < pgSize; i++)
      block[i] = 0;
    }
    usleep(1);
  }

  return 0;
}
```

Запись логов на баше для Windows:

```
#!/bin/bash
while true
do
systeminfo | findstr /C:"Avaliable Physical Memory" >> log.txt
sleep 0.1s
done
```

2. Составить график свободной памяти

График для ОС Linux

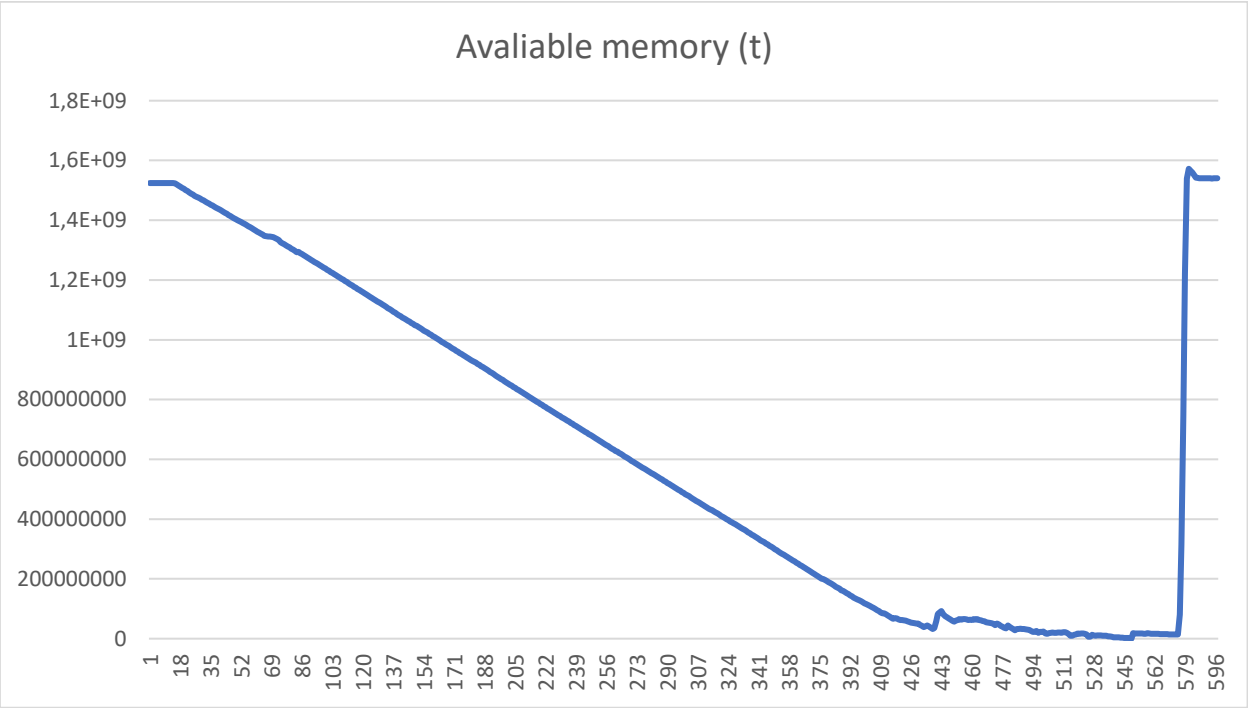
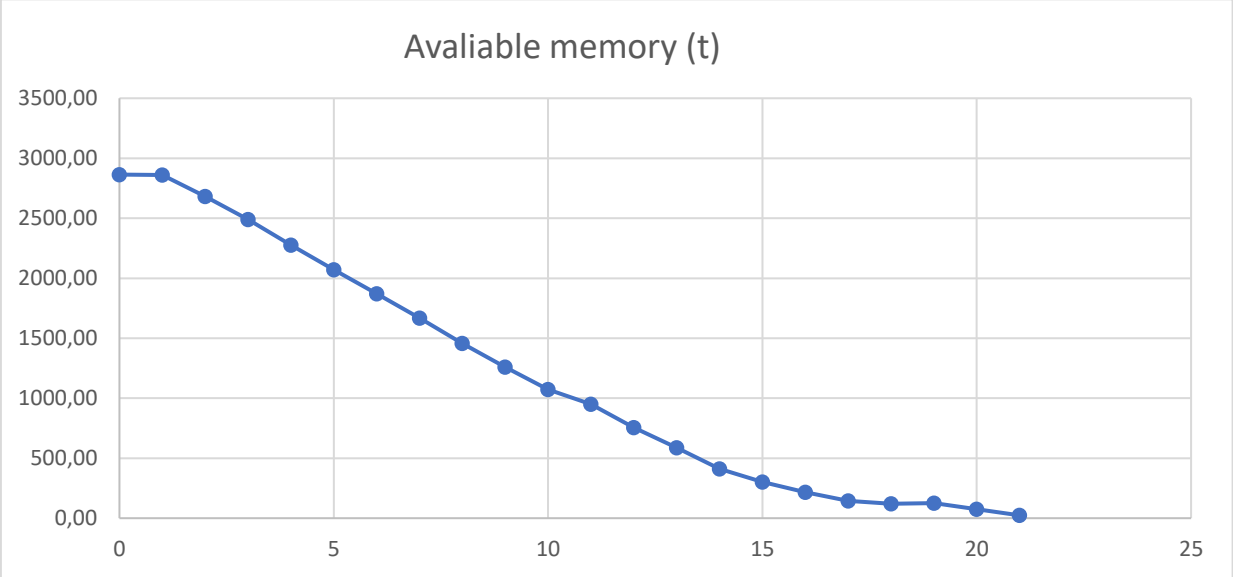


График для ОС Windows



3. Ознакомиться с работой демона OOM Killer в Linux

Принцип работы OOM Killer:

Когда заканчивается память, вызывается функция `out_of_memory()`. В ней есть функция `select_bad_process()`, которая получает оценку от функции `badness()`. Под раздачу попадет самый «плохой» процесс. Функция `badness()` выбирает процесс по определенным правилам.

1. Ядру нужен какой-то минимум памяти для себя.
2. Нужно освободить много памяти.
3. Не нужно завершать процессы, которые используют мало памяти.
4. Нужно завершить минимум процессов.
5. Сложные алгоритмы, которые повышают шансы на завершение для тех процессов, которые пользователь сам хочет завершить.

Выполнив все эти проверки, OOM изучает оценку (`oom_score`). OOM назначает `oom_score` каждому процессу, а потом умножает это значение на объем памяти. У процессов с большими значениями больше шансов стать жертвами OOM Killer. Процессы, связанные с привилегированным пользователем, имеют более низкую оценку и меньше шансов на принудительное завершение.

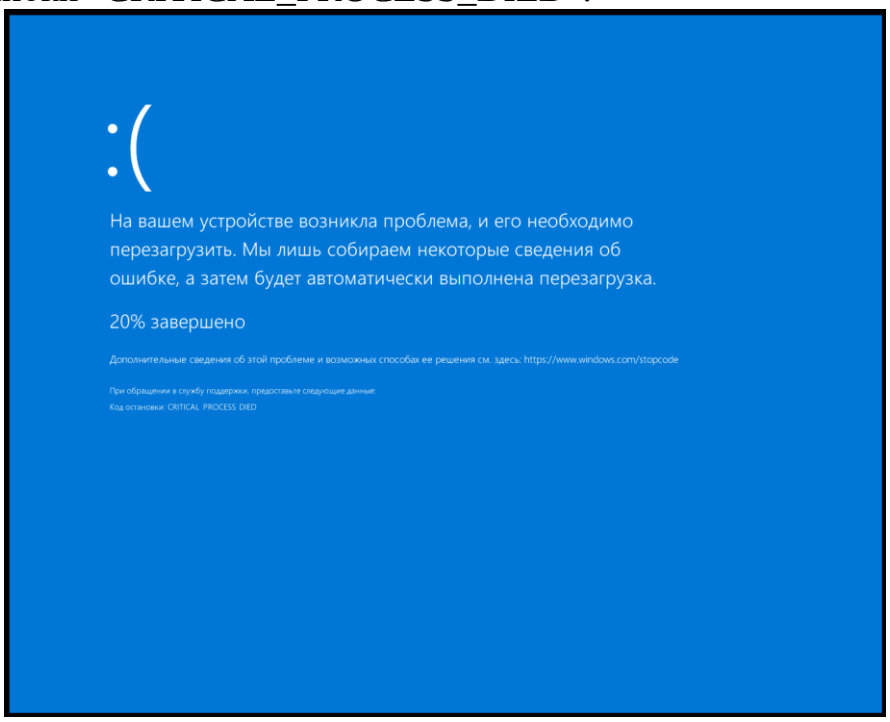
В ходе работы удалось спровоцировать убийство процесса OOM Killer'ом:

```
ildan@vb:~/OS/lab2$ gcc -o f f.c
ildan@vb:~/OS/lab2$ ./f
Killed
```

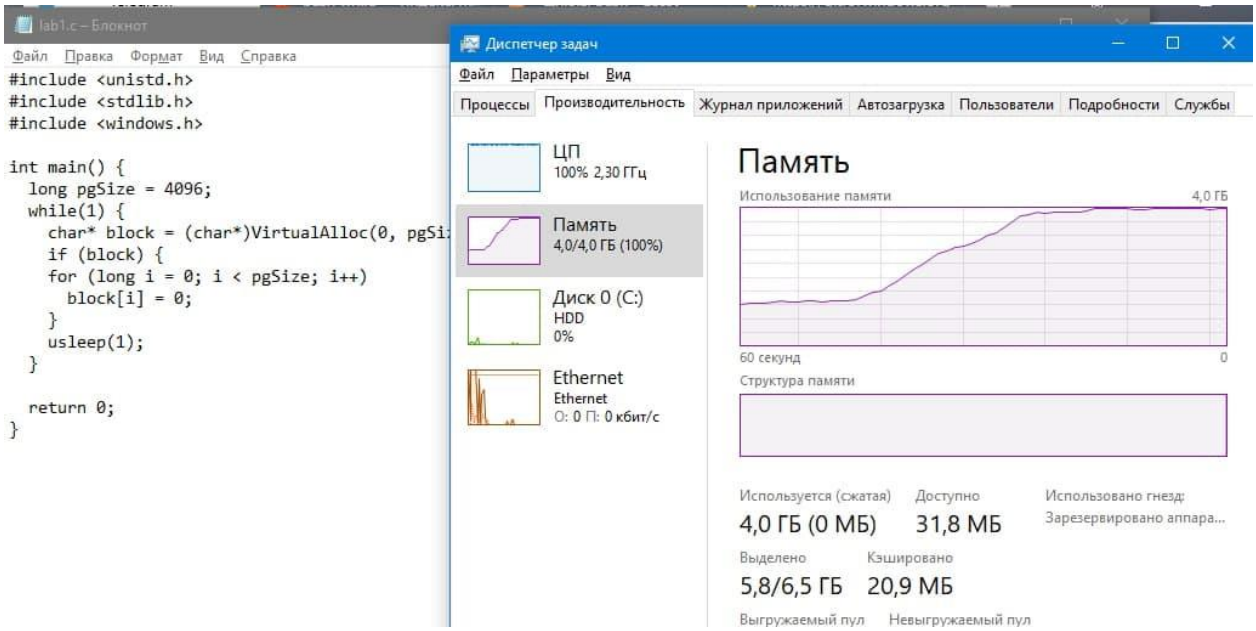
Также это видно на скриншоте выше (*график для ОС Linux*). Приблизительно на 575 тике работы программы, ее убил OOM Killer и вся выделенная программой память очистилась.

4. Достичь сообщения о невозможности выделить память в Windows

После того, как Windows выделила всю оперативную физическую память, она начала резервировать виртуальную память. Суммарно в момент краша ОС было выделенно 11Гб памяти, после чего появился "синий экран смерти" с кодом ошибки "CRITICAL_PROCESS_DIED":



Приложение к лабораторной работе №2:



Также в приложенных файлов представлены логи, записанные для Windows и Linux.