ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Факультет безопасности информационных технологий

Дисциплина:

«Операционные системы»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 2

Membomb

Выполнила:

Студентка гр. №N3253

Пастухова А.А.

Проверил:

Ханов А.Р.

Санкт-Петербург

2022 г.

Задачи:

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

Membomb для Linux

Программа выделения памяти и заполнения ее нулями с шагом, равным размеру страницы памяти для Linux. Также отдельный запуск bash команды, которая каждые 0.1с записывает количество свободной памяти в текстовый файл mem_log.txt.

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

#define TRUE 1
int main(){
   long page_size = sysconf(_SC_PAGESIZE);
   while (TRUE){
    void *ptr = mmap(NULL, page_size, PROT_WRITE|PROT_READ, MAP_PRIVATE| MAP_ANONYMOUS, -1,0);
    usleep(10);
    memset(ptr, 0, page_size);
   }
   return 0;
}
```

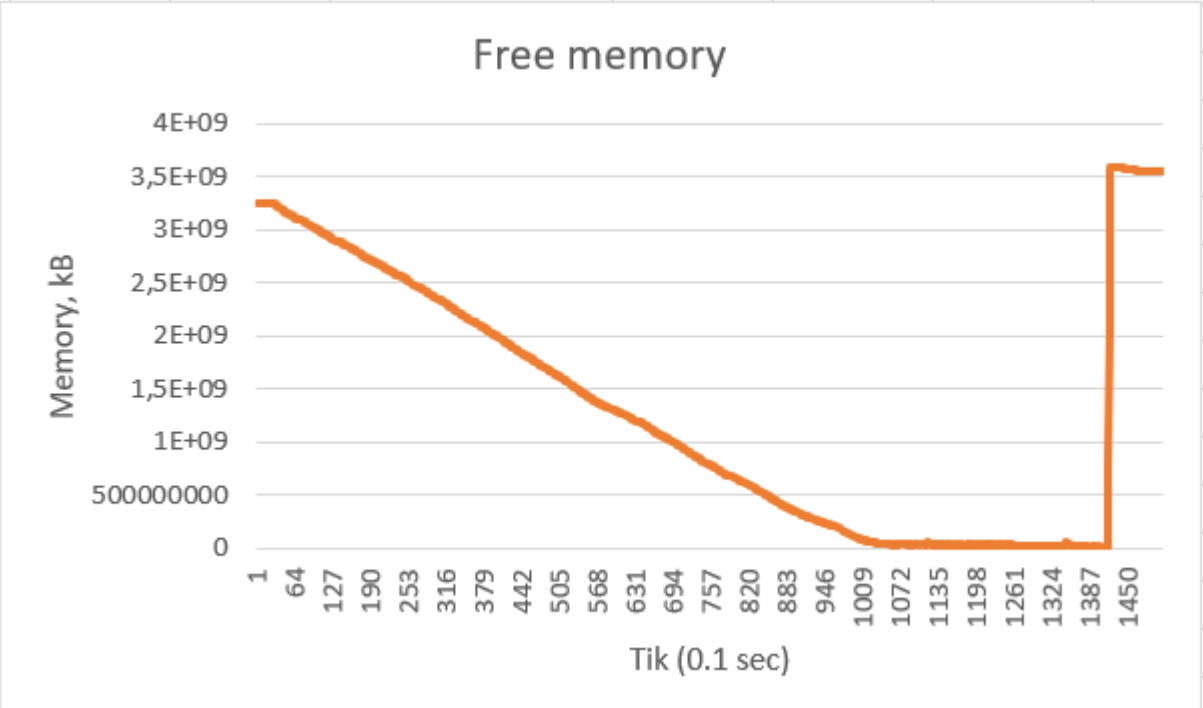
Запись логов на bash

```
#!/bin/bash
free -s 0.1 -b >> mem_log.txt
```

График свободной памяти для ОС Linux

Видно, что на протяжении всего времени работы «бомбы» количество свободной памяти уменьшалось линейно. Примерно на тысячном тике оно достигло своего минимума и через какое-то время произошел резкий скачок, что свидетельствует о срабатывании OOM Killer. Количество свободной

памяти превысило первоначальные показатели, мне кажется, это может быть связано с тем, что сама система даже во время работы бомбы пыталась завершить некоторые другие маловажные процессы. Это видно из графика – при детальном рассмотрении видны некоторые скачки на прямой уменьшения количества памяти.



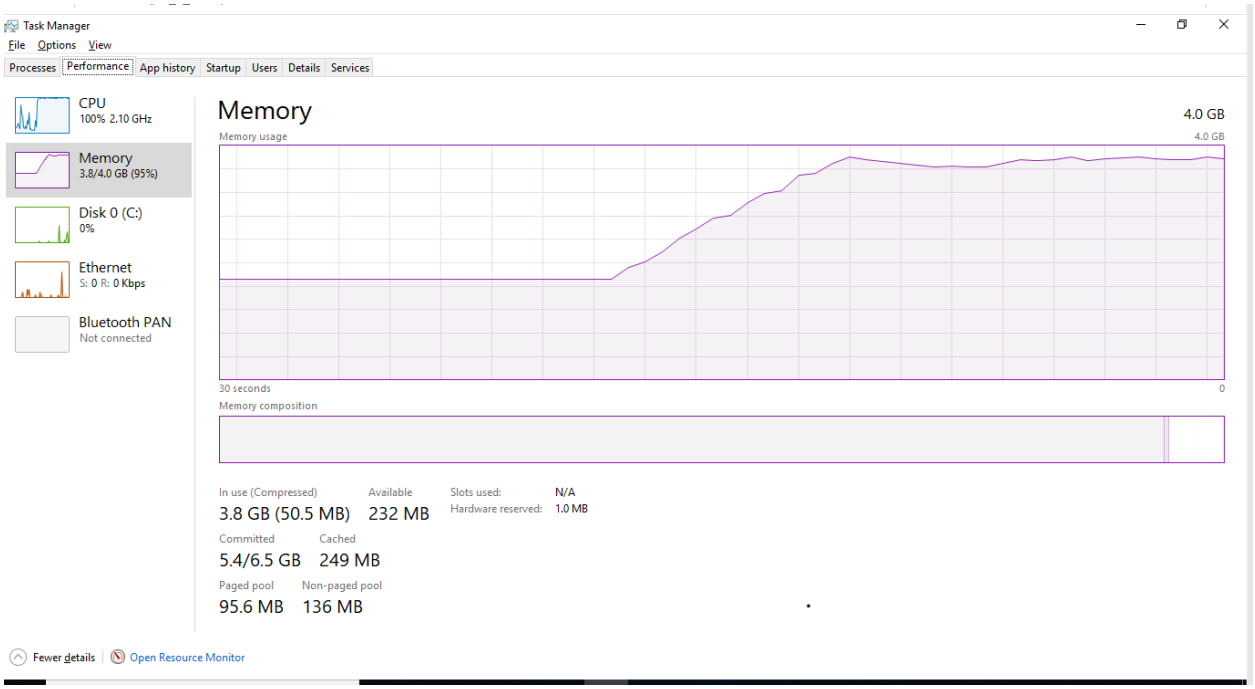
Membomb для Windows

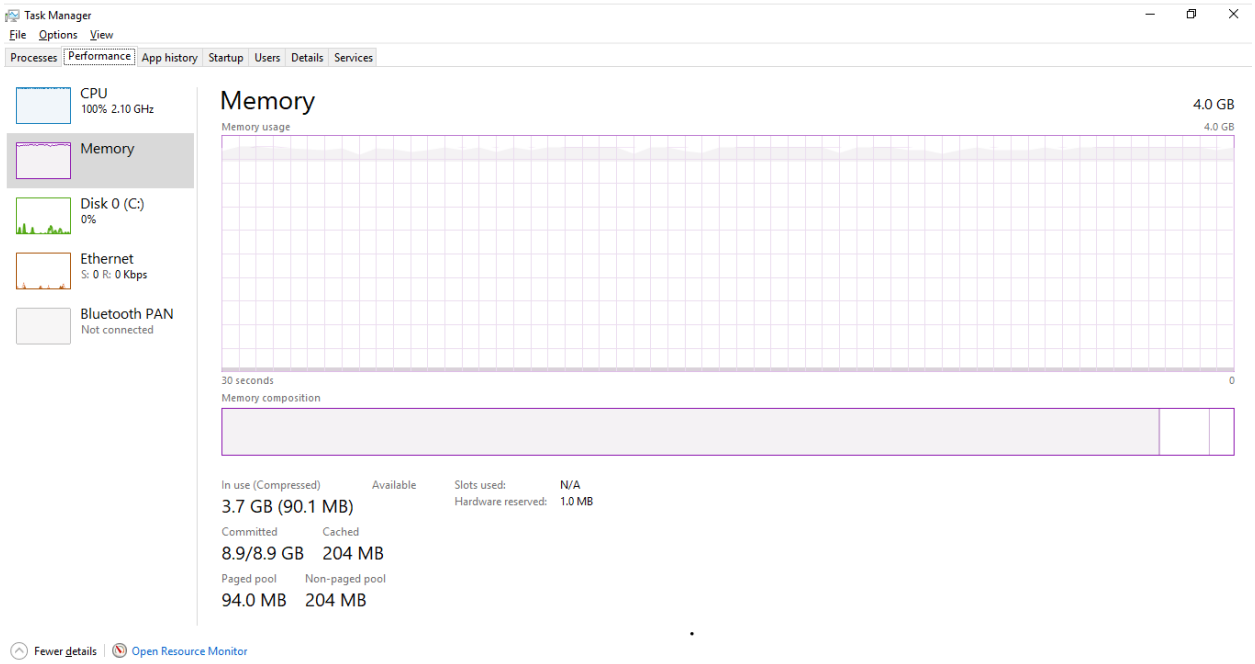
В бесконечном цикле программа выделяет пямять по размеру страницы и заполняет ее нулями, используется команда VirtualAlloc. Логгер для отслеживания памяти такой же, как в Linux версии.

```
#include <unistd.h>
#include <stdlib.h>
#include <windows.h>
int main() {
 SYSTEM_INFO info;
 GetSystemInfo(&info);
 DWORD pgSize = info.dwPageSize;
 while(1) {
 char* block = (char*)VirtualAlloc(0, pgSize, MEM_COMMIT, PAGE_READWRITE);
 if (block) {
 for (long i = 0; i < pgSize; i++)
 block[i] = 0;
 }
```

```
 }
 return 0;
}
```

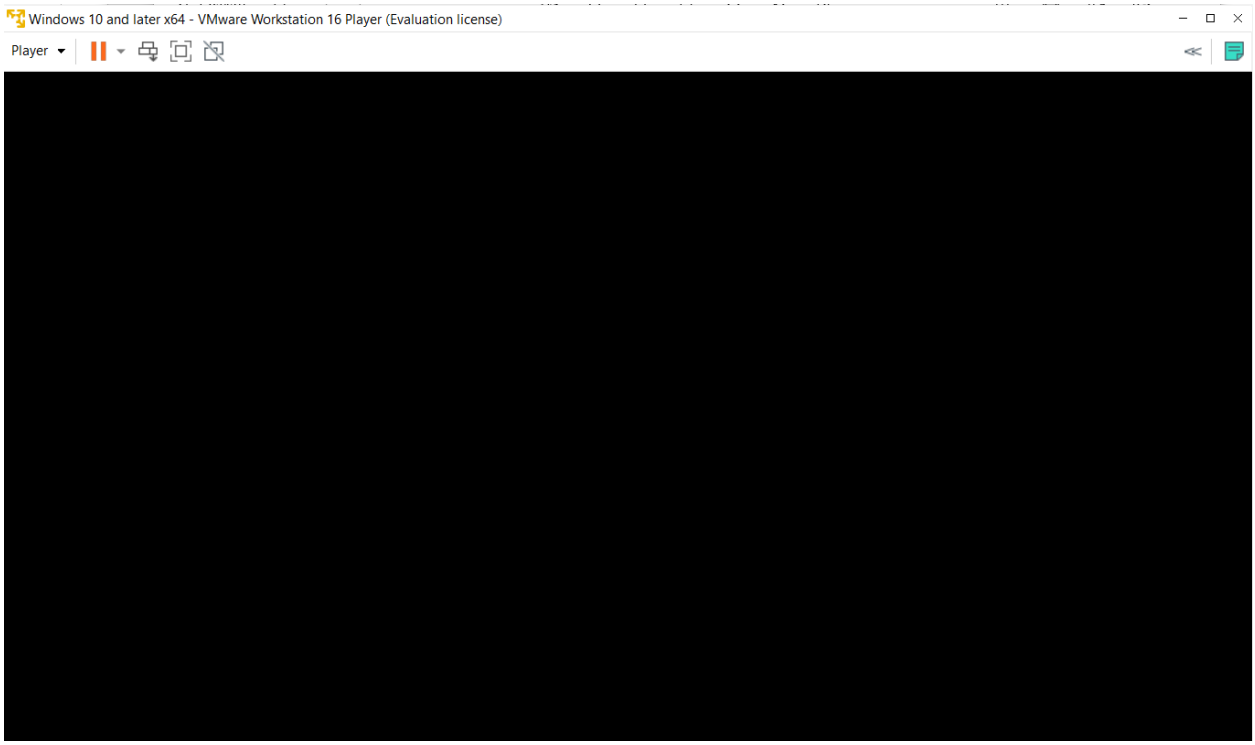
На графиках количества использованной памяти в начальные секунды эксперимента, видно, что сначала заполнилась вся память для виртуальной машины, а на другом графике уже заполнялась основная память.





Даже при многократном запуске программ корректно записывать логги в отдельный файл не получалось, так как операционная система слишком медленно реагировала, а при запуске бомбы прекращала выполнение других

программ, в том числе и логгера, чтобы выделить все ресурсы на выделение памяти и заполнения ее нулями.



Я запускала membomb на виртуальной машине Windows, где ресурсы были ограничены, поэтому вместо синего экрана смерти системы зависла, не реагировала, показала черный экран, после чего мне пришлось ее перезагрузить.

OOM Killer

OOM Killer — это способ ядра решить проблему, когда памяти недостаточно. Известно, что виртуальной памяти может быть бесконечно много (в пределах адресации), а вот физической — вполне конечное число. Иногда процессы системы съедают ее всю, и системе надо кого-то убить, чтобы продолжить работу. Текущая реализация OOM Killer в Linux стремится выбрать наименее важный процесс. Он выбирает среди всех процессов, кроме init и kernel threads, самый негодный (badness).

Для того, чтобы завершить процесс ОС вызывает функцию **out_of_memory**. После чего OOM Killer решает какой именно процесс ему завершить и потом проделывает это с ним.

Перед тем как вызвать out_of_memory выполняются некоторые проверки, которые помогают рассчитать очки негодности процесса (**badness ponts**):

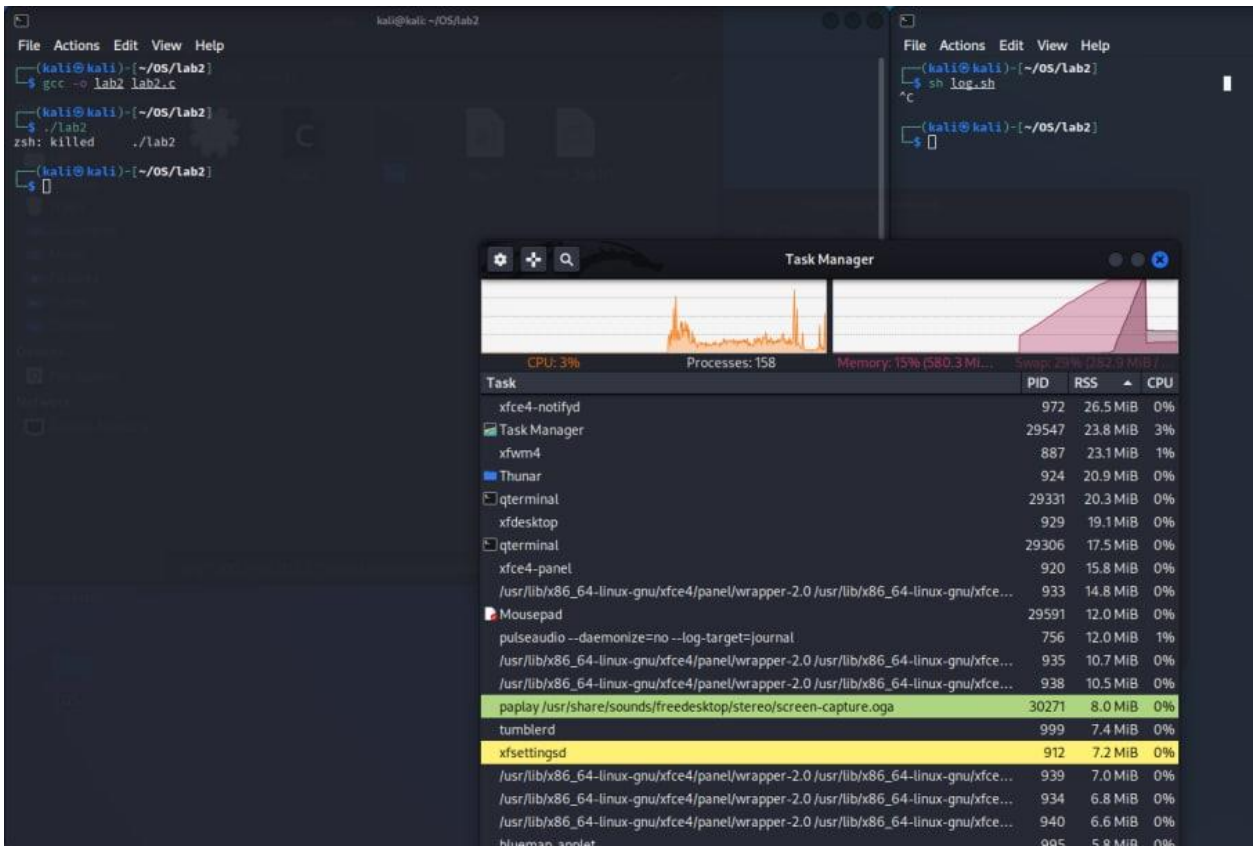
Достаточно ли в ОС еще места подкачки (swap) (nr_swap_pages > 0)? Если да, то не вызываем OOM

Был ли завершений процесс в течение последних 5 секунд? Если да, то не вызываем OOM

И т.д.

По итогам этого всего алгоритма самый негодный (с самыми большими очками) процесс будет убит.

Бывает, необходимо защитить определенный, важные процессы от "убиения". В конце подсчета очков для процесса, число возводится в степень приоритета, которая может принимать значения -15...15 или -17. "-17" означает, что процесс не будет принудительно завершен вообще. Значение приоритета хранится в файлике /proc/$PID/oom_adj. Соответственно, дабы защитить процесс от "убиения", необходимо узнать PID этого процесса и записать в его файл oom_adj значение -17.



На скриншоте показаны запуск membomb и как OOM Killer «убивает» самый ресурснозатратный процесс, особенно хорошо видно это на графиках плнировщика задач.

swapon устанавливает область файла или блочного устройства для подкачки данных и присваивает ей имя path. **swapoff** прекращает подкачку данных в файл или блочное устройство path. Команда **swapoff -a** отключает swap.

Помощь и консультации в выполнении работы оказывал **Шарифуллин И.А.**