Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Факультет безопасности информационных технологий

Дисциплина:

“Операционные системы”

Лабораторная работа №1

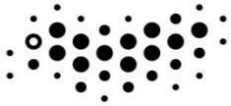
“Fork Bomb”

Выполнил:

ст. группы N3246 Цыдыпов А.О.

Проверил:

Ханов А.Р.



Санкт-Петербург 2022

Задание:

1. Написать программу forkbomb для Linux, Windows.
2. Составить график числа процессов в ОС.
3. Проанализировать, как ОС реагирует на forkbomb.

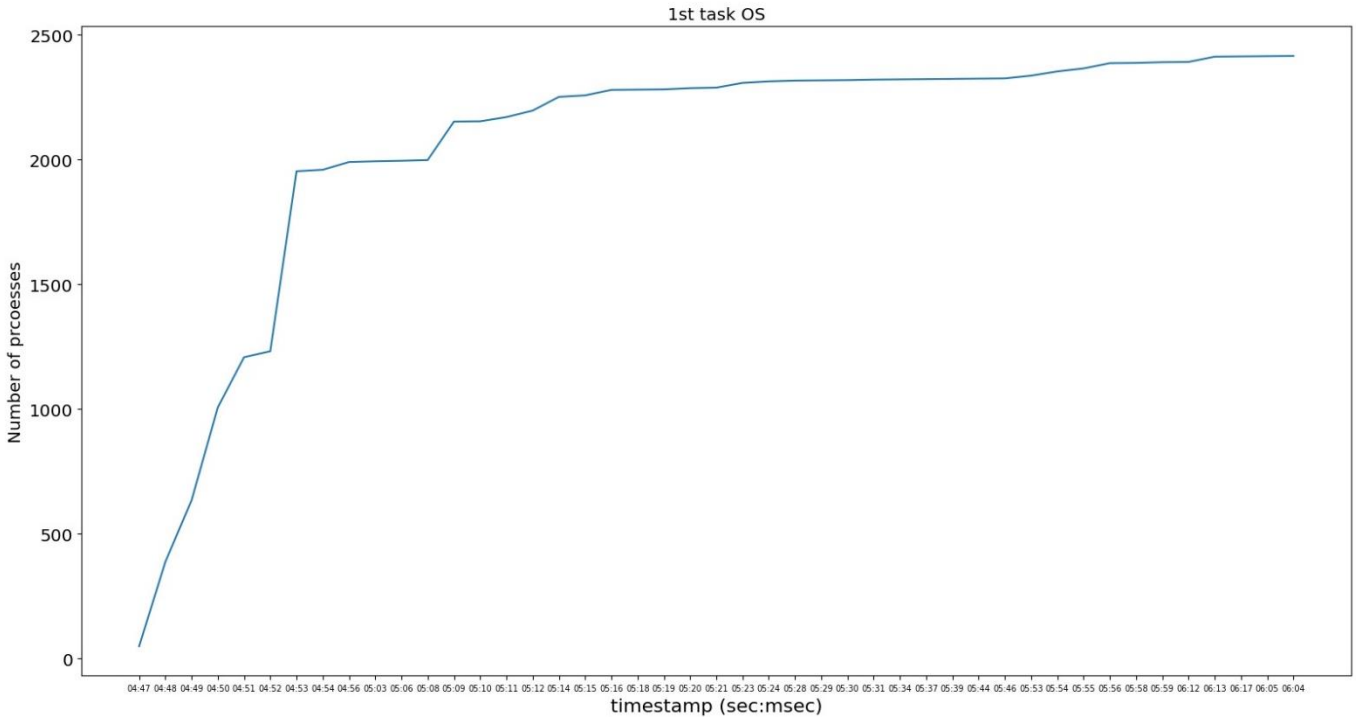
Linux:

Fork Bomb для ОС Linux была написана на скриптовом языке bash.

Код программы forkBomb:

```
1. #!/bin/bash
2. while true
3. do
4. ./forkBomb & date +"%T" >> log
5. done
```

График числа процессов:



Анализ реакции ОС:

По графику видно, что спустя некоторое время скорость создания новых процессов сильно падает. Это происходит, потому что ОС недостаточно ресурсов после запуска около 2000 одинаковых процессов. Все мощности процессора начинают быть задействованы на 100%.

Windows:

Код программы forkBomb.cpp:

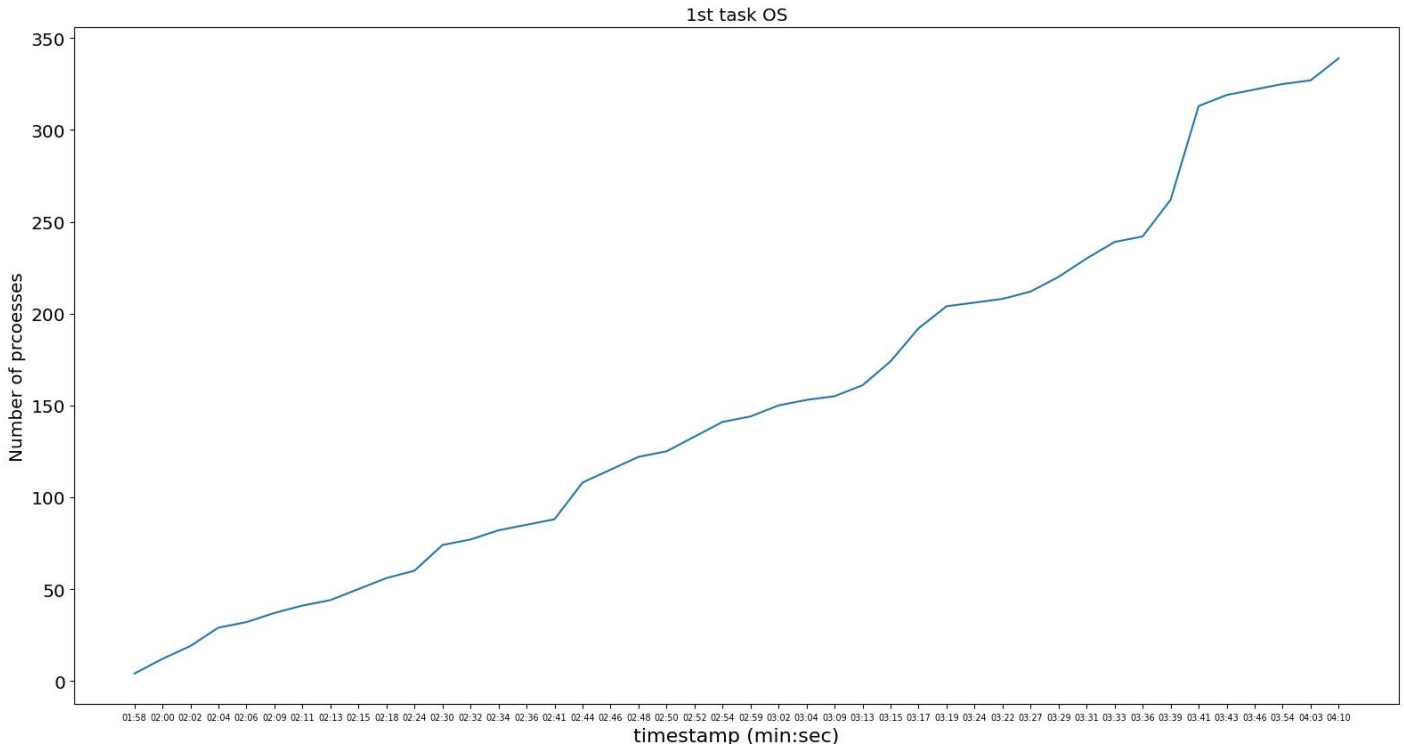
```cpp
#include <windows.h>
#include <stdio.h>
#include <tchar.h>
#include <iostream>
#include <string>
#include <time.h>
#include <chrono>
#include <ctime>

using namespace std;

void slice_str(const char* str, char* buffer, size_t start, size_t end)
{
    size_t j = 0;
    for (size_t i = start; i <= end; ++i) {
        buffer[j++] = str[i];
    }
    buffer[j] = 0;
}

int main(int argc, TCHAR* argv[])
{
    int c = 0;
    while (1) {
        system("start forkBomb.exe");
        //Loging
        if (c == 0) {
            FILE* f = fopen("log.csv", "a");
            char timestamp[64];
            auto end = std::chrono::system_clock::now();
            std::time_t end_time = std::chrono::system_clock::to_time_t(end);
            slice_str(std::ctime(&end_time), timestamp, 11, 19);
            fprintf(f, "%s\n", timestamp);
            fclose(f);
            c = 1;
        }
    }
    return 0;
```

График числа процессов:



Анализ реакции ОС:

Windows справился гораздо хуже чем Linux Ubuntu 20.04. Здесь колличество процессов увеличивается практически линейно, хоть и достигает в пике всего лишь 350 одновременно открытых процессов. Если в Linux было возможно хоть как-то избавиться от fork bomb и вернуть систему в нормальное состояние, то в ОС Windows 10 после первых 2 секунд система полностью перестает реагировать на действия пользователя и спустя какое-то время просто перезагружается.

Код, использованный для построения графиков (python):

Linux version:

```
1. import csv
2. import matplotlib.pyplot as plt
3.
4. params = {'legend.fontsize': 'x-large',
5.           'figure.figsize': (15, 5),
6.          'axes.labelsize': 'x-large',
7.          'axes.titlesize':'x-large',
8.          'xtick.labelsize':'x-small',
9.          'ytick.labelsize':'x-large'}
10. plt.rcParams.update(params)
11.
12. logs = []
13. with open('log.csv', 'r') as fd:
14.   reader = csv.reader(fd)
15.   for row in reader:
16.           logs.append(row[0])
17.
18. count = 0
19. timestamps = []
20.
21. for el in logs:
22.   if(el not in timestamps):
23.           timestamps.append(el)
24.
25. procCounter = [0]*len(timestamps)
26. for i in range(len(timestamps)):
27.   for log in logs:
28.           if(log==timestamps[i]):
29.                   count+=1
30.   procCounter[i] = count
31. print(procCounter)
32.
33. newTimestamps = []
34. for timestamp in timestamps:
35.   print(timestamp[3:8])
36.   newTimestamps.append(timestamp[3:8])
37.
38.
39.
40. plt.plot(newTimestamps, procCounter)
41. plt.xlabel("timestamp (sec:msec)", fontsize=16)
42. plt.ylabel("Number of prcoesses")
43.
44. plt.title("1st task OS")
45.
46. plt.show()
```

Windows version:

```
1. import csv
2. import matplotlib.pyplot as plt
3.
4. params = {'legend.fontsize': 'x-large',
5.           'figure.figsize': (15, 5),
6.          'axes.labelsize': 'x-large',
7.          'axes.titlesize':'x-large',
8.          'xtick.labelsize':'x-small',
9.          'ytick.labelsize':'x-large'}
10. plt.rcParams.update(params)
11.
12. logs = []
13. with open('log.csv', 'r') as fd:
14.   reader = csv.reader(fd)
15.   for row in reader:
16.           logs.append(row[0])
17.
18. count = 0
19. timestamps = []
20.
21. for el in logs:
22.   if(el not in timestamps):
23.           timestamps.append(el)
24.
25. procCounter = [0]*len(timestamps)
26. for i in range(len(timestamps)):
27.   for log in logs:
28.           if(log==timestamps[i]):
29.                   count+=1
30.   procCounter[i] = count
31. print(procCounter)
32.
33. newTimestamps = []
34. for timestamp in timestamps:
35.   print(timestamp[3:8])
36.   newTimestamps.append(timestamp[3:8])
37.
38. x = []
39. y = []
40. for i in range(len(newTimestamps)):
41.   if(i%2==0):
42.           x.append(newTimestamps[i])
43.
44. for i in range(len(procCounter)):
45.   if(i%2==0):
46.           y.append(procCounter[i])
47.
48. plt.plot(x, y)
49. plt.xlabel("timestamp (min:sec)", fontsize=16)
50. plt.ylabel("Number of prcoesses")
51.
52. plt.title("1st task OS")
53.
54. plt.show()
```