

Факультет программ инженерии

Лабораторная работа #1
Операционные системы

Выполнил: Куприянов А.А

Санкт-Петербург, 2020

Вариант:

A=118;B=0xB44603B3;C=mmap;D=84;E=45;F=nocache;G=11;H=seq;I=13;J=sum;K=futex

```
1 #define ALLOC_ADDR      0xB44603B3
2 #define THREADS_AMOUNT  3
3 #define FILL_MEM_SIZE    118
4 #define DUMP_FILE_SIZE   45
5 #define DATA_BLOCK_SIZE 11
6 #define COUNTERS_THREAD_AMOUNT 11
```

Заполнение памяти случайными данными:

```
void * mmapAddr;
mmapAddr = mmap(addr, size, PROT_READ | PROT_WRITE, MAP_ANONYMOUS | MAP_PRIVATE, -1, 0);

for (int i = 0; i < THREADS_AMOUNT; i++) {

    arg = malloc(sizeof(arg));
    arg->size = size / THREADS_AMOUNT;
    arg->writeAddr = mmapAddr + i * ((arg->size / 8) + (arg->size % 8 > 0 ? 1 : 0));
    arg->dataSource = fopen("/dev/urandom", "r");
    arg->logEnabled = logEnabled;
    err = pthread_create(&threads[i], NULL, threadFunc, arg);

    if (err != 0)
    {
        puts("Error with creating thread");
        exit(err);
    }
}

for (int i = 0; i < THREADS_AMOUNT; i++) {
    err = pthread_join(threads[i], NULL);

    if (err != 0)
    {
        exit(err);
    }
}
```

Запись данных с памяти в файл (no-cache):

```
void
dumpMem(int fd, void * addr, int size, int * futex) {
    wait_on_futex_value(futex, 0);
    *futex = 0;
    int iterations = size / DATA_BLOCK_SIZE;
    int res = ftruncate(fd, 0);
    for (int i = 0; i < iterations; i++) {
        void * pointer = addr + i;
        size_t r = write(fd, &pointer, DATA_BLOCK_SIZE);
    }

    *futex = 1;
}
```

Чтение и агрегация файла:

```
int * futex = args->futex;
wait_on_futex_value(futex, 0);
*futex = 0;
uint64_t sum = 0;

if (lseek(args->fd, 0, SEEK_SET) == -1) {
    return -1;
}

unsigned char buf [DATA_BLOCK_SIZE];

while(1) {

    size_t readBytes = read(args->fd, &buf, DATA_BLOCK_SIZE);

    if (readBytes == -1) {
        perror("Error file read");
        break;
    }

    if (readBytes < DATA_BLOCK_SIZE) {
        if (readBytes == 0) {
            break;
        }
    } else {
        for (int i = 0; i < DATA_BLOCK_SIZE; i++) {
            sum += buf[i];
        }
    }
}

*futex = 1;

return 0;
```

Снятие блокировки со всех потоков:

```
for (int i = 0; i < filesAmount; i++) {
    int * futex = args->dumpMap[i]->futex;
    wake_futex_blocking(futex, 1);
}
```

Снятие значений памяти

До аллокации

	total	used	free	shared	buff/cache	available
Mem:	16692132	8103976	8358804	17720	229352	8454424
Swap:	50331648	10240	50321408			

После аллокации

	total	used	free	shared	buff/cache	available
Mem:	16692132	8099780	8363000	17720	229352	8458620
Swap:	50331648	10240	50321408			

После заполнения данными

	total	used	free	shared	buff/cache	available
Mem:	16692132	8146868	8315912	17720	229352	8411532
Swap:	50331648	10240	50321408			

После деаллокации

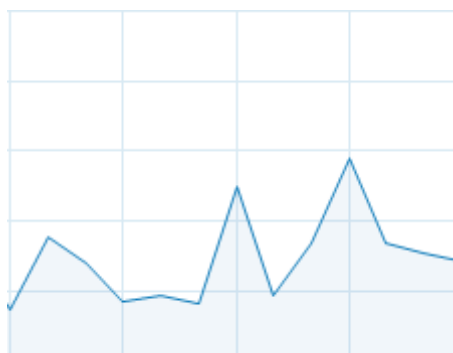
	total	used	free	shared	buff/cache	available
Mem:	16692132	8097920	8364860	17720	229352	8460480
Swap:	50331648	10240	50321408			

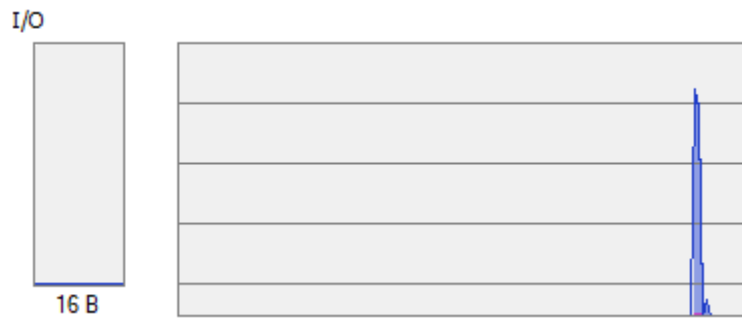
```
mmap(0xb44603b3, 123731968, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0xb4460000
```

Запись в памяти в отдельных потоках:

Filling address: 0xb494aaab - pid: 1079	tid: 140164705552128
Filling address: 0xb4460000 - pid: 1079	tid: 140164714006272
Filling address: 0xb4e35556 - pid: 1079	tid: 140164697097984

Показания процессора





Показания vmstat:

```
procs -----memory----- ---swap-- ----io----- -system-- -----cpu-----
r  b   swpd   free   buff  cache   si   so    bi    bo    in   cs  us  sy  id  wa  st
1  0   36284 331332  41460 543408    2    6   535  1331   309  462   6  13  76   4   0
```

Файлы дампов:

```
-rw----- 1 apploid apploid 47185138 Oct 29 19:40 dump.0
-rw----- 1 apploid apploid 47185138 Oct 29 19:40 dump.1
-rw----- 1 apploid apploid 47185138 Oct 29 19:40 dump.2
```

Синхронизация потоков осуществляется через futex:

```
void wait_on_futex_value(int* futex_addr, int val) {
    futex(futex_addr, FUTEX_WAIT, val);
}

void wake_futex_blocking(int* futex_addr, int val) {
    while (1) {
        int futex_rc = futex(futex_addr, FUTEX_WAKE, val);
        if (futex_rc == -1) {
            perror("futex wake");
            exit(1);
        } else if (futex_rc > 0) {
            return;
        }
    }
}
```

Часть вывода rтар

```
00000000b4460000      4      4      4 rw--- [ anon ]
000055f8de847000      4      4      0 r---- main.o
000055f8de848000      8      8      0 r-x-- main.o
000055f8de84a000      4      4      0 r---- main.o
000055f8de84b000      4      4      4 r---- main.o
000055f8de84c000      4      4      4 rw--- main.o
000055f8df874000    132      8      8 rw--- [ anon ]
00007fb57effe000      4      0      0 ----- [ anon ]
.
.
.
00007fb591956000      4      4      4 rw-s- [ shmid=0x14 ]
00007fb591957000      4      4      4 r---- ld-2.31.so
00007fb591958000      4      4      4 rw--- ld-2.31.so
00007fb591959000      4      4      4 rw--- [ anon ]
00007fff9a5d7000    132     20     20 rw--- [ stack ]
00007fff9a5f8000     12      0      0 r---- [ anon ]
00007fff9a5fb000      4      4      0 r-x-- [ anon ]
fffffffffff6000000      4      0      0 --x-- [ anon ]
```

Также снимались показания через htop:

```
 1 [|||||||||||||||||||||||||||||||||100.0%] Tasks: 33, 22 thr; 2 running
 2 [|||||||||||||||||||||||||||||||||100.0%] Load average: 0.92 0.91 0.80
Mem[|||||||||||||||||||||||||||||79.2M/981M] Uptime: 01:10:01
Swp[|||||||||25.2M/112M]
```

```
21185 root      20    0 292M 2316 2100 S 108.  0.2 1:22.53 ./main.o
```

Вывод:

В этой лабораторной работе я ознакомился не только с утилитами для мониторинга, но и самим языком Си. Особенно интересной частью было использование потоков и передача аргументов через указатели. Язык Си оказался очень мощным инструментом для создания разных утилит и програм нуждающихся в более близкой работе с ситемными вызовами и самим "железом"

Неожиданно я также столкнулся проблемой установки stap на Windows. Казалось бы, WSL должен был уместно работать с ним, но оказалось нет. Дело в том, что stap искал linux-headers, linux-image и dbgsym в них с версией (uname -r). При запуске этой команды в WSL выходит: 4.4.0-19041-Microsoft.

Разумеется, для него нету каких-либо утилит. Попытка установить отличную от него версию и использование символической ссылки привели к краху и трате впустую 7 гб пространства на диске для установки dbgsym

Поэтому я решил сделать все это на удаленной машине с Ubuntu 20. Оставалось только добавить репозитории и установить dbgsym - и ... stap заработал!