# PA 1: Address Spaces and Resource Usage Monitoring

Leran Ma and Sishi Cheng

February 16, 2020

1.

(a) Addresses 1 are for the local variables; Addresses 2 are for the dynamically allocated variables.

We know that local variables are stored in the stack, and dynamically allocated variables are stored in the heap. After each invocation of the function, the sizes of the stack and the heap grow bigger. Also, we know that the stack grows down towards smaller addresses, and the heap grows up towards the larger address. According to the console output, Address 1 is decreasing, and Address 2 is increasing. Thus, Address 1 matches the stack, and Address 2 matches the heap.

```
e5-cse-204-07.cse.psu.edu 41$ ./prog1
Address 1 = 0x7ffc6ee1b5c0    Address 2 = 0x19d6010
Address 1 = 0x7ffc6edeb020    Address 2 = 0x19d6210
Address 1 = 0x7ffc6edbaa80    Address 2 = 0x19d6410
Address 1 = 0x7ffc6ed8a4e0    Address 2 = 0x19d6610
Address 1 = 0x7ffc6ed59f40    Address 2 = 0x19d6810
Address 1 = 0x7ffc6ed299a0    Address 2 = 0x19d6a10
Address 1 = 0x7ffc6ecf9400    Address 2 = 0x19d6c10
Address 1 = 0x7ffc6ecc8e60    Address 2 = 0x19d6e10
Address 1 = 0x7ffc6ec988c0    Address 2 = 0x19d7010
Address 1 = 0x7ffc6ec68320    Address 2 = 0x19d7210
Enter any key to exit
```

On our system, the stack grows down towards smaller addresses, and the heap grows up towards the larger address.

(b) The size of the process stack when it is waiting for user input is 2140 kB.

(c) The size of the process heap when it is waiting for user input 132 kB.

(d) The address limits of the stack are `0x7ffc6ec37000-0x7ffc6ee4e000`. The address limits of the heap are `0x019d6000-0x019f7000`.

According to the console output, Addresses 1 are within the address limits of the stack. Addresses 2 are within the address limits of the heap.

(e) execve() executes prog1.
brk() adjusts the size of the data segment of the address space for prog1.
mmap() serves the memory allocation purpose.
access() checks if the user is permitted to execute the file prog1.
open() opens required libraries for prog1 execution.
stat() checks the status of the required library.
fstat() checks the status of a previously opened file.
mprotect() protects a specified region of memory from unexpected modification.
close() close an opened file.
arch_prctl() set the state of the current thread.
munmap() serves the memory deallocation purpose.
write() displays the output of prog1.
read() reads in user inputs.
exit_group() exits all threads of prog1.

2.

(a) Size of the code in bytes:
32-bit:
(i) 1520
(ii) 2008K
(iii) 1996K

64-bit:
(i) 1722
(ii) 4088K
(iii) 4072K

(b) Prog2_32 received a segmentation fault after executing "c = malloc (30000);"

```
Breakpoint 1, allocate (count=4) at prog2.c:11
11          c = malloc (30000);
(gdb) n

Program received signal SIGSEGV, Segmentation fault.
allocate (count=4) at prog2.c:11
11          c = malloc (30000);
(gdb)
```

Prog2_64 received a segmentation fault after executing "allocate(count - 1);"

```
Breakpoint 1, allocate (count=5) at prog2.c:13
13          allocate(count - 1);
(gdb) n

Program received signal SIGSEGV, Segmentation fault.
0x0000000000400618 in allocate (count=<error reading variable: Cannot access memory at address 0x7fffff7fafac>) at prog2.c:5
5       {
(gdb)
```

(c) According to the result of the command "`cat /proc/PID/limits`" for prog2_32, the limit of the stack size in our system is 8388608 bytes.

```
e5-cse-204-07.cse.psu.edu 33$ cat /proc/18451/limits
Limit                     Soft Limit           Hard Limit           Units
Max cpu time              unlimited            unlimited            seconds
Max file size             unlimited            unlimited            bytes
Max data size             unlimited            unlimited            bytes
Max stack size            8388608              unlimited            bytes
Max core file size        0                    unlimited            bytes
Max resident set          unlimited            unlimited            bytes
Max processes             4096                 63051                processes
Max open files            1024                 4096                 files
Max locked memory         65536                65536                bytes
Max address space         unlimited            unlimited            bytes
Max file locks            unlimited            unlimited            locks
Max pending signals       63051                63051                signals
Max msgqueue size         819200               819200               bytes
Max nice priority         0                    0
Max realtime priority     0                    0
Max realtime timeout      unlimited            unlimited            us
```

According to the result of "`cat /proc/PID/maps`" for prog2_32, the bottom of the stack is at the address `0xffffe000`.

```
e5-cse-204-07.cse.psu.edu 32$ cat /proc/18451/maps
08048000-08049000 r-xp 00000000 00:35 9183504                            /home/ugrads/lkm5463/cmpsc473/cmpsc473-project1-473_pa1_lm_sc/prog2/prog2_32
08049000-0804a000 r--p 00000000 00:35 9183504                            /home/ugrads/lkm5463/cmpsc473/cmpsc473-project1-473_pa1_lm_sc/prog2/prog2_32
0804a000-0804b000 rw-p 00001000 00:35 9183504                            /home/ugrads/lkm5463/cmpsc473/cmpsc473-project1-473_pa1_lm_sc/prog2/prog2_32
0804b000-08097000 rw-p 00000000 00:00 0                                  [heap]
f7dda000-f7ddb000 rw-p 00000000 00:00 0
f7ddb000-f7f9f000 r-xp 00000000 fd:00 805679868                          /usr/lib/libc-2.17.so
f7f9f000-f7fa0000 ---p 001c4000 fd:00 805679868                          /usr/lib/libc-2.17.so
f7fa0000-f7fa2000 r--p 001c4000 fd:00 805679868                          /usr/lib/libc-2.17.so
f7fa2000-f7fa3000 rw-p 001c6000 fd:00 805679868                          /usr/lib/libc-2.17.so
f7fa3000-f7fa6000 rw-p 00000000 00:00 0
f7fd7000-f7fd9000 rw-p 00000000 00:00 0
f7fd9000-f7fda000 r-xp 00000000 00:00 0                                  [vdso]
f7fda000-f7ffc000 r-xp 00000000 fd:00 805361642                          /usr/lib/ld-2.17.so
f7ffc000-f7ffd000 r--p 00021000 fd:00 805361642                          /usr/lib/ld-2.17.so
f7ffd000-f7ffe000 rw-p 00022000 fd:00 805361642                          /usr/lib/ld-2.17.so
ff91e000-ffffe000 rw-p 00000000 00:00 0                                  [stack]
```

Thus, the top of the stack cannot exceed `0xffffe000`-8388608, which is `0xff7fe000`. According to the output of prog2_32, we can safely print the stack address for 6 times, and we get a segmentation fault when we try the 7th time. And from the change of the stack address, we expect that the address printed at the 7th time would be lower than `0xff7fe000`. Therefore, it is reasonable to say that the segmentation fault is caused by stack overflow.

```
Starting program: /home/ugrads/lkm5463/cmpsc473/cmpsc473-project1-473_pa1_lm_sc/prog2/prog2_32
Entering Function

Stack Address  = 0xffed7f9c    Heap Address = 0x804b008

Stack Address  = 0xffdb2fec    Heap Address = 0x8052540

Stack Address  = 0xffc8e03c    Heap Address = 0x8059a78

Stack Address  = 0xffb6908c    Heap Address = 0x8060fb0

Stack Address  = 0xffa440dc    Heap Address = 0x80684e8

Stack Address  = 0xff91f12c    Heap Address = 0x806fa20


Program received signal SIGSEGV, Segmentation fault.
```

Similarly, for 64 bits, the limit of stack is still 8388608 bytes.

```
e5-cse-204-07.cse.psu.edu 39$ cat /proc/20180/limits
Limit                     Soft Limit           Hard Limit           Units
Max cpu time              unlimited            unlimited            seconds
Max file size             unlimited            unlimited            bytes
Max data size             unlimited            unlimited            bytes
Max stack size            8388608              unlimited            bytes
Max core file size        0                    unlimited            bytes
Max resident set          unlimited            unlimited            bytes
Max processes             4096                 63051                processes
Max open files            1024                 4096                 files
Max locked memory         65536                65536                bytes
Max address space         unlimited            unlimited            bytes
Max file locks            unlimited            unlimited            locks
Max pending signals       63051                63051                signals
Max msgqueue size         819200               819200               bytes
Max nice priority         0                    0
Max realtime priority     0                    0
Max realtime timeout      unlimited            unlimited            us
```

The bottom of the stack is at `0x7fffffff000`, and thus the top of the stack cannot be lower than `0x7fffff7ff000` (`0x7fffffff000`-8388608).

```
e5-cse-204-07.cse.psu.edu 38$ cat /proc/20180/maps
00400000-00401000 r-xp 00000000 00:35 9183505                            /home/ugrads/lkm5463/cmpsc473/cmpsc473-project1-473_pa1_lm_sc/prog2/prog2_64
00600000-00601000 r--p 00000000 00:35 9183505                            /home/ugrads/lkm5463/cmpsc473/cmpsc473-project1-473_pa1_lm_sc/prog2/prog2_64
00601000-00602000 rw-p 00001000 00:35 9183505                            /home/ugrads/lkm5463/cmpsc473/cmpsc473-project1-473_pa1_lm_sc/prog2/prog2_64
00602000-0064e000 rw-p 00000000 00:00 0                                  [heap]
7ffff7a0d000-7ffff7bd1000 r-xp 00000000 fd:00 537242972                  /usr/lib64/libc-2.17.so
7ffff7bd1000-7ffff7dd0000 ---p 001c4000 fd:00 537242972                  /usr/lib64/libc-2.17.so
7ffff7dd0000-7ffff7dd4000 r--p 001c3000 fd:00 537242972                  /usr/lib64/libc-2.17.so
7ffff7dd4000-7ffff7dd6000 rw-p 001c7000 fd:00 537242972                  /usr/lib64/libc-2.17.so
7ffff7dd6000-7ffff7ddb000 rw-p 00000000 00:00 0
7ffff7ddb000-7ffff7dfd000 r-xp 00000000 fd:00 537241598                  /usr/lib64/ld-2.17.so
7ffff7fc4000-7ffff7fc7000 rw-p 00000000 00:00 0
7ffff7ff8000-7ffff7ffa000 rw-p 00000000 00:00 0
7ffff7ffa000-7ffff7ffc000 r-xp 00000000 00:00 0                          [vdso]
7ffff7ffc000-7ffff7ffd000 r--p 00021000 fd:00 537241598                  /usr/lib64/ld-2.17.so
7ffff7ffd000-7ffff7ffe000 rw-p 00022000 fd:00 537241598                  /usr/lib64/ld-2.17.so
7ffff7ffe000-7ffff7fff000 rw-p 00000000 00:00 0
7fffff91f000-7ffffffff000 rw-p 00000000 00:00 0                          [stack]
ffffffffff600000-ffffffffff601000 r-xp 00000000 00:00 0                  [vsyscall]
```

Again, from the change of the 6 printed stack addresses, we expect that the 7th stack address is probably lower than `0x7fffff7ff000`. Therefore, stack overflow is the cause of the segmentation fault.

```
Starting program: /home/ugrads/lkm5463/cmpsc473/cmpsc473-project1-473_pa1_lm_sc/prog2/prog2_64
Entering Function

Stack Address  = 0x7fffffed8dd0    Heap Address = 0x602010

Stack Address  = 0x7fffffdb3e20    Heap Address = 0x609550

Stack Address  = 0x7fffffc8ee70    Heap Address = 0x610a90

Stack Address  = 0x7fffffb69ec0    Heap Address = 0x617fd0

Stack Address  = 0x7fffffa44f10    Heap Address = 0x61f510

Stack Address  = 0x7fffff91ff60    Heap Address = 0x626a50


Program received signal SIGSEGV, Segmentation fault.
```

(d) In prog2_32, there are already 8 frames in the stack.

```
#0  allocate (count=4) at prog2.c:11
#1  0x080484e6 in allocate (count=5) at prog2.c:13
#2  0x080484e6 in allocate (count=6) at prog2.c:13
#3  0x080484e6 in allocate (count=7) at prog2.c:13
#4  0x080484e6 in allocate (count=8) at prog2.c:13
#5  0x080484e6 in allocate (count=9) at prog2.c:13
#6  0x080484e6 in allocate (count=10) at prog2.c:13
#7  0x08048511 in main (argc=1, argv=0xffffd014) at prog2.c:21
```

The address of the frames: (using the "info f" command)
#0 0xff91f130
#1 0xffa440e0
#2 0xffb69090
#3 0xffc8e040
#4 0xffdb2ff0
#5 0xffed7fa0
#6 0xffffcf50
#7 0xffffcf80

The size of each frame (in bytes):
#1 1200048 (= 0xffa440e0-0xff91f130)
#2 1200048 (= 0xffb69090-0xffa440e0)
#3 1200048 (= 0xffc8e040-0xffb69090)
#4 1200048 (= 0xffdb2ff0-0xffc8e040)
#5 1200048 (= 0xffed7fa0-0xffdb2ff0)
#6 1200048 (= 0xffffcf50-0xffed7fa0)
#7 48 (= 0xffffcf80-0xffffcf50)

8388608/1200048=6.99. Therefore, 6 invocations of the recursive function should be possible on my system.

When we actually execute the program, 6 invocations of the recursive function occur without any error. The 7th invocation causes a segmentation fault.

In prog2_64, there are already 8 frames in the stack.

```
#0  0x0000000000400618 in allocate (count=<error reading variable: Cannot access memory at address 0x7fffff7fafac>) at prog2.c:5
#1  0x0000000000400662 in allocate (count=5) at prog2.c:13
#2  0x0000000000400662 in allocate (count=6) at prog2.c:13
#3  0x0000000000400662 in allocate (count=7) at prog2.c:13
#4  0x0000000000400662 in allocate (count=8) at prog2.c:13
#5  0x0000000000400662 in allocate (count=9) at prog2.c:13
#6  0x0000000000400662 in allocate (count=10) at prog2.c:13
#7  0x000000000040068e in main (argc=1, argv=0x7fffffffde78) at prog2.c:21
```

The address of the frames:
#0 0x7fffff91ff50
#1 0x7fffffa44f00
#2 0x7fffffb69eb0
#3 0x7fffffc8ee60
#4 0x7fffffdb3e10
#5 0x7fffffed8dc0
#6 0x7fffffffdd70
#7 0x7fffffffdda0

The size of each frame (in bytes):
#1 1200048 (= 0x7fffffa44f00-0x7fffff91ff50)
#2 1200048 (= 0x7fffffb69eb0-0x7fffffa44f00)
#3 1200048 (= 0x7fffffc8ee60-0x7fffffb69eb0)
#4 1200048 (= 0x7fffffdb3e10-0x7fffffc8ee60)
#5 1200048 (= 0x7fffffed8dc0-0x7fffffdb3e10)
#6 1200048 (= 0x7fffffffdd70-0x7fffffed8dc0)
#7 48 (= 0x7fffffffdda0-0x7fffffffdd70)

8388608/1200048=6.99. Therefore, 6 invocations of the recursive function should be possible on my system.

When we actually execute the program, 6 invocations of the recursive function occur without any error. The 7th invocation causes a segmentation fault.

(e) In general, the content of a frame includes return address, arguments, local variables, frame pointer, and saved registers.

For both prog_32, contents present in a frame of the recursive function and their sizes:

- Return address: 4 bytes
- Arguments:
  - count: 4 bytes
- Local variables:
  - x: 1200000 bytes
  - c: 4 bytes
- frame pointer: 4 bytes

For prog_64, contents present in a frame of the recursive function and their sizes:

Return address: 8 bytes
Arguments:
count: 4 bytes
Local variables:
x: 1200000 bytes
c: 8 bytes
frame pointer: 8 bytes

3.

(a) 32-bit (in bytes):
(i) 1747
(ii) 2268K
(iii) 2256K

64-bit (in bytes):
(i) 2017
(ii) 66424K
(iii) 66408K

(b) The program statement causing the segmentation fault is “memset(ch1,'*',sizeof(b[0])+i);”. The memset function attempts to access some address outside the heap. This is caused by heap overflow.

According to the result of the command “cat /proc/PID/maps” for prog3_32, the top of the heap (f7d99000) reaches the address space for shared libraries. Thus, any further access will cause a segmentation fault.

```
e5-cse-204-10.cse.psu.edu 20$ cat /proc/27547/maps
08048000-08049000 r-xp 00000000 00:2f 3030507                            /home/ugrads/smc6823/cmpsc473-proj
ect1-473_pa1_lm_sc/prog3/prog3_32
08049000-0804a000 r--p 00000000 00:2f 3030507                            /home/ugrads/smc6823/cmpsc473-proj
ect1-473_pa1_lm_sc/prog3/prog3_32
0804a000-0804b000 rw-p 00001000 00:2f 3030507                            /home/ugrads/smc6823/cmpsc473-proj
ect1-473_pa1_lm_sc/prog3/prog3_32
0804b000-0806c000 rw-p 00000000 00:00 0                                  [heap]
dd400000-dd421000 rw-p 00000000 00:00 0
dd421000-dd500000 ---p 00000000 00:00 0
dd5c6000-f7d99000 rw-p 00000000 00:00 0
f7d99000-f7f5d000 r-xp 00000000 fd:00 805679742                          /usr/lib/libc-2.17.so
f7f5d000-f7f5e000 ---p 001c4000 fd:00 805679742                          /usr/lib/libc-2.17.so
f7f5e000-f7f60000 r--p 001c4000 fd:00 805679742                          /usr/lib/libc-2.17.so
f7f60000-f7f61000 rw-p 001c6000 fd:00 805679742                          /usr/lib/libc-2.17.so
f7f61000-f7f64000 rw-p 00000000 00:00 0
f7f64000-f7fa4000 r-xp 00000000 fd:00 805679750                          /usr/lib/libm-2.17.so
f7fa4000-f7fa5000 r--p 0003f000 fd:00 805679750                          /usr/lib/libm-2.17.so
f7fa5000-f7fa6000 rw-p 00040000 fd:00 805679750                          /usr/lib/libm-2.17.so
f7fd7000-f7fd9000 rw-p 00000000 00:00 0
f7fd9000-f7fda000 r-xp 00000000 00:00 0                                  [vdso]
f7fda000-f7ffc000 r-xp 00000000 fd:00 806231790                          /usr/lib/ld-2.17.so
f7ffc000-f7ffd000 r--p 00021000 fd:00 806231790                          /usr/lib/ld-2.17.so
f7ffd000-f7ffe000 rw-p 00022000 fd:00 806231790                          /usr/lib/ld-2.17.so
fffdd000-ffffe000 rw-p 00000000 00:00 0                                  [stack]
```

Also, for prog3_64, according to the result of the command “cat /proc/PID/maps”, the top of the heap (7ffff770b000) reaches the address space for shared libraries. Thus, any further access will cause a segmentation fault.

```
e5-cse-204-07.cse.psu.edu 85$ cat /proc/21512/maps
00400000-00401000 r-xp 00000000 00:35 9185260                            /home/ugrads/lkm5463/cmpsc473/cmpsc473-project1-473_pa1_lm_sc/prog3/prog3_64
00600000-00601000 r--p 00000000 00:35 9185260                            /home/ugrads/lkm5463/cmpsc473/cmpsc473-project1-473_pa1_lm_sc/prog3/prog3_64
00601000-00602000 rw-p 00001000 00:35 9185260                            /home/ugrads/lkm5463/cmpsc473/cmpsc473-project1-473_pa1_lm_sc/prog3/prog3_64
00602000-00623000 rw-p 00000000 00:00 0                                  [heap]
7ffee8000000-7ffee8021000 rw-p 00000000 00:00 0
7ffee8021000-7ffeec000000 ---p 00000000 00:00 0
7ffeee8e8000-7ffff770b000 rw-p 00000000 00:00 0
7ffff770b000-7ffff78cf000 r-xp 00000000 fd:00 537242972                  /usr/lib64/libc-2.17.so
7ffff78cf000-7ffff7ace000 ---p 001c4000 fd:00 537242972                  /usr/lib64/libc-2.17.so
7ffff7ace000-7ffff7ad2000 r--p 001c3000 fd:00 537242972                  /usr/lib64/libc-2.17.so
7ffff7ad2000-7ffff7ad4000 rw-p 001c7000 fd:00 537242972                  /usr/lib64/libc-2.17.so
7ffff7ad4000-7ffff7ad9000 rw-p 00000000 00:00 0
7ffff7ad9000-7ffff7bda000 r-xp 00000000 fd:00 537243112                  /usr/lib64/libm-2.17.so
7ffff7bda000-7ffff7dd9000 ---p 00101000 fd:00 537243112                  /usr/lib64/libm-2.17.so
7ffff7dd9000-7ffff7dda000 r--p 00100000 fd:00 537243112                  /usr/lib64/libm-2.17.so
7ffff7dda000-7ffff7ddb000 rw-p 00101000 fd:00 537243112                  /usr/lib64/libm-2.17.so
7ffff7ddb000-7ffff7dfd000 r-xp 00000000 fd:00 537241598                  /usr/lib64/ld-2.17.so
7ffff7f62000-7ffff7fc7000 rw-p 00000000 00:00 0
7ffff7ff8000-7ffff7ffa000 rw-p 00000000 00:00 0
7ffff7ffa000-7ffff7ffc000 r-xp 00000000 00:00 0                          [vdso]
7ffff7ffc000-7ffff7ffd000 r--p 00021000 fd:00 537241598                  /usr/lib64/ld-2.17.so
7ffff7ffd000-7ffff7ffe000 rw-p 00022000 fd:00 537241598                  /usr/lib64/ld-2.17.so
7ffff7ffe000-7ffff7fff000 rw-p 00000000 00:00 0
7ffffffde000-7ffffffff000 rw-p 00000000 00:00 0                          [stack]
ffffffffff600000-ffffffffff601000 r-xp 00000000 00:00 0                  [vsyscall]
```

(c) The error in prog2 is caused by stack overflow, whereas the error in prog3 is caused by heap overflow.

4.

(a) Because the dynamically allocated memories are freed conditionally in the function allocate, not all memories are freed. This leads to serious memory leaks.

This error can be fixed by removing the if condition in the function allocate.

(b) (i) user CPU time used: about 1100 ms

(ii) system CPU time used: about 480 ms
The difference between user CPU time and system CPU time is the modes. User CPU time is the execution time spent in user mode, but the system CPU time is the execution time spent in kernel mode.

(iii) maximum resident set size: 1814736 KB
The upper bound of the physical memory that the process occupied during the whole execution process.

(iii) signals received: 0
User processes and the OS may send signals.

(iv) voluntary context switches: 3

(v) involuntary context switches: 32
According to the *Linux Programmer's Manual*, voluntary context switches happen when "a process voluntarily giving up the processor before its time slice was completed (usually to await availability of a resource)." Involuntary context switches happen when "a higher priority process becoming runnable or because the current process exceeded its time slice."

5.

(a) (i) The PID of the parent and child processes are different.
(ii) The addresses of the dynamic variables are the same.

Screenshot of prog71 execution:

```
e5-cse-204-07.cse.psu.edu 54$ ./prog71
PID of prog71.c: 7454
PID: 7454, Address pointed by p: 0x1a77010
PID: 7455, Address pointed by p: 0x1a77010
Hello, We are in child process (process B), PID: 7455
Hello, We are in prog72.c , PID: 7455
PID: 7455, Address pointed by p: 0x1d35010
Hello, We are in parent process (process A), PID: 7454
Sending signal to process B (PID: 7455)
Hello, We are in prog72's signal handler , PID: 7455
Signal Received from process A, and process B is killed
Welcome back to process A, PID: 7454
```

(b) Address space of the parent process before the "execv" command:

```
e5-cse-204-07.cse.psu.edu 26$ pmap 3706
3706:   /home/ugrads/lkm5463/cmpsc473/cmpsc473-project1-473_pa1_lm_sc/prog7/prog71
0000000000400000      4K r-x-- prog71
0000000000600000      4K r---- prog71
0000000000601000      4K rw--- prog71
0000000000602000    132K rw---   [ anon ]
00007ffff7a0d000   1808K r-x-- libc-2.17.so
00007ffff7bd1000   2044K ----- libc-2.17.so
00007ffff7dd0000     16K r---- libc-2.17.so
00007ffff7dd4000      8K rw--- libc-2.17.so
00007ffff7dd6000     20K rw---   [ anon ]
00007ffff7ddb000    136K r-x-- ld-2.17.so
00007ffff7fc4000     12K rw---   [ anon ]
00007ffff7ff8000      8K rw---   [ anon ]
00007ffff7ffa000      8K r-x--   [ anon ]
00007ffff7ffc000      4K r---- ld-2.17.so
00007ffff7ffd000      4K rw--- ld-2.17.so
00007ffff7ffe000      4K rw---   [ anon ]
00007ffffffde000    132K rw---   [ stack ]
ffffffffff600000      4K r-x--   [ anon ]
 total             4352K
e5-cse-204-07.cse.psu.edu 27$ cat /proc/3706/maps
00400000-00401000 r-xp 00000000 00:30 9186142                            /home/ugrads/lkm5463/cmpsc473/cmpsc473-project1-473_pa1_lm_sc/prog7/prog71
00600000-00601000 r--p 00000000 00:30 9186142                            /home/ugrads/lkm5463/cmpsc473/cmpsc473-project1-473_pa1_lm_sc/prog7/prog71
00601000-00602000 rw-p 00001000 00:30 9186142                            /home/ugrads/lkm5463/cmpsc473/cmpsc473-project1-473_pa1_lm_sc/prog7/prog71
00602000-00623000 rw-p 00000000 00:00 0                                  [heap]
7ffff7a0d000-7ffff7bd1000 r-xp 00000000 fd:00 537242972                  /usr/lib64/libc-2.17.so
7ffff7bd1000-7ffff7dd0000 ---p 001c4000 fd:00 537242972                  /usr/lib64/libc-2.17.so
7ffff7dd0000-7ffff7dd4000 r--p 001c3000 fd:00 537242972                  /usr/lib64/libc-2.17.so
7ffff7dd4000-7ffff7dd6000 rw-p 001c7000 fd:00 537242972                  /usr/lib64/libc-2.17.so
7ffff7dd6000-7ffff7ddb000 rw-p 00000000 00:00 0
7ffff7ddb000-7ffff7dfd000 r-xp 00000000 fd:00 537241598                  /usr/lib64/ld-2.17.so
7ffff7fc4000-7ffff7fc7000 rw-p 00000000 00:00 0
7ffff7ff8000-7ffff7ffa000 rw-p 00000000 00:00 0
7ffff7ffa000-7ffff7ffc000 r-xp 00000000 00:00 0                          [vdso]
7ffff7ffc000-7ffff7ffd000 r--p 00021000 fd:00 537241598                  /usr/lib64/ld-2.17.so
7ffff7ffd000-7ffff7ffe000 rw-p 00022000 fd:00 537241598                  /usr/lib64/ld-2.17.so
7ffff7ffe000-7ffff7fff000 rw-p 00000000 00:00 0
7ffffffde000-7ffffffff000 rw-p 00000000 00:00 0                          [stack]
ffffffffff600000-ffffffffff601000 r-xp 00000000 00:00 0                  [vsyscall]
```

Address space of the child process before the “execv” command:

```
e5-cse-204-07.cse.psu.edu 28$ pmap 3725
3725:   /home/ugrads/lkm5463/cmpsc473/cmpsc473-project1-473_pa1_lm_sc/prog7/prog71
0000000000400000      4K r-x-- prog71
0000000000600000      4K r---- prog71
0000000000601000      4K rw--- prog71
0000000000602000    132K rw---   [ anon ]
00007ffff7a0d000   1808K r-x-- libc-2.17.so
00007ffff7bd1000   2044K ----- libc-2.17.so
00007ffff7dd0000     16K r---- libc-2.17.so
00007ffff7dd4000      8K rw--- libc-2.17.so
00007ffff7dd6000     20K rw---   [ anon ]
00007ffff7ddb000    136K r-x-- ld-2.17.so
00007ffff7fc4000     12K rw---   [ anon ]
00007ffff7ff8000      8K rw---   [ anon ]
00007ffff7ffa000      8K r-x--   [ anon ]
00007ffff7ffc000      4K r---- ld-2.17.so
00007ffff7ffd000      4K rw--- ld-2.17.so
00007ffff7ffe000      4K rw---   [ anon ]
00007fffffffde000    132K rw---   [ stack ]
ffffffffff600000      4K r-x--   [ anon ]
 total             4352K
e5-cse-204-07.cse.psu.edu 29$ cat /proc/3725/maps
00400000-00401000 r-xp 00000000 00:30 9186142                            /home/ugrads/lkm5463/cmpsc473/cmpsc473-project1-473_pa1_lm_sc/prog7/prog71
00600000-00601000 r--p 00000000 00:30 9186142                            /home/ugrads/lkm5463/cmpsc473/cmpsc473-project1-473_pa1_lm_sc/prog7/prog71
00601000-00602000 rw-p 00001000 00:30 9186142                            /home/ugrads/lkm5463/cmpsc473/cmpsc473-project1-473_pa1_lm_sc/prog7/prog71
00602000-00623000 rw-p 00000000 00:00 0                                  [heap]
7ffff7a0d000-7ffff7bd1000 r-xp 00000000 fd:00 537242972                  /usr/lib64/libc-2.17.so
7ffff7bd1000-7ffff7dd0000 ---p 001c4000 fd:00 537242972                  /usr/lib64/libc-2.17.so
7ffff7dd0000-7ffff7dd4000 r--p 001c3000 fd:00 537242972                  /usr/lib64/libc-2.17.so
7ffff7dd4000-7ffff7dd6000 rw-p 001c7000 fd:00 537242972                  /usr/lib64/libc-2.17.so
7ffff7dd6000-7ffff7ddb000 rw-p 00000000 00:00 0
7ffff7ddb000-7ffff7dfd000 r-xp 00000000 fd:00 537241598                  /usr/lib64/ld-2.17.so
7ffff7fc4000-7ffff7fc7000 rw-p 00000000 00:00 0
7ffff7ff8000-7ffff7ffa000 rw-p 00000000 00:00 0
7ffff7ffa000-7ffff7ffc000 r-xp 00000000 00:00 0                          [vdso]
7ffff7ffc000-7ffff7ffd000 r--p 00021000 fd:00 537241598                  /usr/lib64/ld-2.17.so
7ffff7ffd000-7ffff7ffe000 rw-p 00022000 fd:00 537241598                  /usr/lib64/ld-2.17.so
7ffff7ffe000-7ffff7fff000 rw-p 00000000 00:00 0
7ffffffde000-7fffffffd000 rw-p 00000000 00:00 0                          [stack]
ffffffffff600000-ffffffffff601000 r-xp 00000000 00:00 0                  [vsyscall]
```

Address space of the parent process after the “execv” command:

```
e5-cse-204-07.cse.psu.edu 33$ pmap 3706
3706:   /home/ugrads/lkm5463/cmpsc473/cmpsc473-project1-473_pa1_lm_sc/prog7/prog71
0000000000400000      4K r-x-- prog71
0000000000600000      4K r---- prog71
0000000000601000      4K rw--- prog71
0000000000602000    132K rw---   [ anon ]
00007ffff7a0d000   1808K r-x-- libc-2.17.so
00007ffff7bd1000   2044K ----- libc-2.17.so
00007ffff7dd0000     16K r---- libc-2.17.so
00007ffff7dd4000      8K rw--- libc-2.17.so
00007ffff7dd6000     20K rw---   [ anon ]
00007ffff7ddb000    136K r-x-- ld-2.17.so
00007ffff7fc4000     12K rw---   [ anon ]
00007ffff7ff8000      8K rw---   [ anon ]
00007ffff7ffa000      8K r-x--   [ anon ]
00007ffff7ffc000      4K r---- ld-2.17.so
00007ffff7ffd000      4K rw--- ld-2.17.so
00007ffff7ffe000      4K rw---   [ anon ]
00007fffffffde000    132K rw---   [ stack ]
ffffffffff600000      4K r-x--   [ anon ]
 total             4352K
e5-cse-204-07.cse.psu.edu 34$ cat /proc/3706/maps
00400000-00401000 r-xp 00000000 00:30 9186142                            /home/ugrads/lkm5463/cmpsc473/cmpsc473-project1-473_pa1_lm_sc/prog7/prog71
00600000-00601000 r--p 00000000 00:30 9186142                            /home/ugrads/lkm5463/cmpsc473/cmpsc473-project1-473_pa1_lm_sc/prog7/prog71
00601000-00602000 rw-p 00001000 00:30 9186142                            /home/ugrads/lkm5463/cmpsc473/cmpsc473-project1-473_pa1_lm_sc/prog7/prog71
00602000-00623000 rw-p 00000000 00:00 0                                  [heap]
7ffff7a0d000-7ffff7bd1000 r-xp 00000000 fd:00 537242972                  /usr/lib64/libc-2.17.so
7ffff7bd1000-7ffff7dd0000 ---p 001c4000 fd:00 537242972                  /usr/lib64/libc-2.17.so
7ffff7dd0000-7ffff7dd4000 r--p 001c3000 fd:00 537242972                  /usr/lib64/libc-2.17.so
7ffff7dd4000-7ffff7dd6000 rw-p 001c7000 fd:00 537242972                  /usr/lib64/libc-2.17.so
7ffff7dd6000-7ffff7ddb000 rw-p 00000000 00:00 0
7ffff7ddb000-7ffff7dfd000 r-xp 00000000 fd:00 537241598                  /usr/lib64/ld-2.17.so
7ffff7fc4000-7ffff7fc7000 rw-p 00000000 00:00 0
7ffff7ff8000-7ffff7ffa000 rw-p 00000000 00:00 0
7ffff7ffa000-7ffff7ffc000 r-xp 00000000 00:00 0                          [vdso]
7ffff7ffc000-7ffff7ffd000 r--p 00021000 fd:00 537241598                  /usr/lib64/ld-2.17.so
7ffff7ffd000-7ffff7ffe000 rw-p 00022000 fd:00 537241598                  /usr/lib64/ld-2.17.so
7ffff7ffe000-7ffff7fff000 rw-p 00000000 00:00 0
7ffffffde000-7fffffffd000 rw-p 00000000 00:00 0                          [stack]
ffffffffff600000-ffffffffff601000 r-xp 00000000 00:00 0                  [vsyscall]
```

Address space of the child process after the “execv” command:

```
e5-cse-204-07.cse.psu.edu 35$ pmap 3725
3725:   Hello World
0000000000400000      4K r-x-- prog72
0000000000600000      4K r---- prog72
0000000000601000      4K rw--- prog72
00007ffff7a0d000   1808K r-x-- libc-2.17.so
00007ffff7bd1000   2044K ----- libc-2.17.so
00007ffff7dd0000     16K r---- libc-2.17.so
00007ffff7dd4000      8K rw--- libc-2.17.so
00007ffff7dd6000     20K rw---   [ anon ]
00007ffff7ddb000    136K r-x-- ld-2.17.so
00007ffff7fc4000     12K rw---   [ anon ]
00007ffff7ff9000      4K rw---   [ anon ]
00007ffff7ffa000      8K r-x--   [ anon ]
00007ffff7ffc000      4K r---- ld-2.17.so
00007ffff7ffd000      4K rw--- ld-2.17.so
00007ffff7ffe000      4K rw---   [ anon ]
00007ffffffde000    132K rw---   [ stack ]
ffffffffff600000      4K r-x--   [ anon ]
 total             4216K
e5-cse-204-07.cse.psu.edu 36$ cat /proc/3725/maps
00400000-00401000 r-xp 00000000 00:30 9182940                            /home/ugrads/lkm5463/cmpsc473/cmpsc473-project1-473_pa1_lm_sc/prog7/prog72
00600000-00601000 r--p 00000000 00:30 9182940                            /home/ugrads/lkm5463/cmpsc473/cmpsc473-project1-473_pa1_lm_sc/prog7/prog72
00601000-00602000 rw-p 00001000 00:30 9182940                            /home/ugrads/lkm5463/cmpsc473/cmpsc473-project1-473_pa1_lm_sc/prog7/prog72
7ffff7a0d000-7ffff7bd1000 r-xp 00000000 fd:00 537242972                  /usr/lib64/libc-2.17.so
7ffff7bd1000-7ffff7dd0000 ---p 001c4000 fd:00 537242972                  /usr/lib64/libc-2.17.so
7ffff7dd0000-7ffff7dd4000 r--p 001c3000 fd:00 537242972                  /usr/lib64/libc-2.17.so
7ffff7dd4000-7ffff7dd6000 rw-p 001c7000 fd:00 537242972                  /usr/lib64/libc-2.17.so
7ffff7dd6000-7ffff7ddb000 rw-p 00000000 00:00 0
7ffff7ddb000-7ffff7dfd000 r-xp 00000000 fd:00 537241598                  /usr/lib64/ld-2.17.so
7ffff7fc4000-7ffff7fc7000 rw-p 00000000 00:00 0
7ffff7ff9000-7ffff7ffa000 rw-p 00000000 00:00 0
7ffff7ffa000-7ffff7ffc000 r-xp 00000000 00:00 0                          [vdso]
7ffff7ffc000-7ffff7ffd000 r--p 00021000 fd:00 537241598                  /usr/lib64/ld-2.17.so
7ffff7ffd000-7ffff7ffe000 rw-p 00022000 fd:00 537241598                  /usr/lib64/ld-2.17.so
7ffff7ffe000-7ffff7fff000 rw-p 00000000 00:00 0
7ffffffde000-7ffffffff000 rw-p 00000000 00:00 0                          [stack]
ffffffffff600000-ffffffffff601000 r-xp 00000000 00:00 0                  [vsyscall]
```

From the 4 screenshots of address spaces above, we can see that the first 3 address spaces are essentially the same. This is because before the “execv” command, the parent process and the child process both execute prog71, and therefore the first 2 screenshots are similar. Besides, because of the if statement, the parent process will never run the “execv” command. Only the child process will run the “execv” command. Even if the child process run the “execv” command, nothing will happen to the parent process. The address space of the parent process will remain the same. However, because the child process executes prog72, the address space of the child process will change to store information about prog72.

(c) stack of prog72 before signal handling:

```
(gdb) bt
#0  main (argc=2, argv=0x7fffffffdef8) at prog72.c:34
(gdb) info f 0
Stack frame at 0x7fffffffde20:
 rip = 0x4008eb in main (prog72.c:34); saved rip 0x7ffff7a2f555
 source language c.
 Arglist at 0x7fffffffde10, args: argc=2, argv=0x7fffffffdef8
 Locals at 0x7fffffffde10, Previous frame's sp is 0x7fffffffde20
 Saved registers:
  rbp at 0x7fffffffde10, rip at 0x7fffffffde18
```

stack of prog72 after signal handling:

```
(gdb) c
Continuing.
Hello, We are in prog72's signal handler , PID: 3725
Signal Received from process A, and process B is killed
[Inferior 4 (process 3725) exited with code 01]
(gdb) info inferiors
  Num  Description       Executable
* 4    <null>            /home/ugrads/lkm5463/cmpsc473/cmpsc473-project1-473_pa1_lm_sc/prog7/prog72
  3    process 3706      /home/ugrads/lkm5463/cmpsc473/cmpsc473-project1-473_pa1_lm_sc/prog7/prog71
  1    <null>            /home/ugrads/lkm5463/cmpsc473/cmpsc473-project1-473_pa1_lm_sc/prog7/prog71
(gdb) bt
No stack.
```

(In the screenshot above, inferior 3 is the parent process, and inferior 4 is the child process. Inferior 1 is just some process generated in previous tests, and it is already terminated. Please ignore inferior 1.)

Before signal handling, the child process (prog72) has one frame in the stack, which is the frame of the main function. After signal handling, the child process is killed. Therefore, its stack does not exist.