Eric Samuel Huddleston

5/7/2021

CS 4853 Section 001

My first step in this assignment was writing the system call code getpages.c, which I implemented almost verbatim from the Assignment 4 explanation video, except adding comments where they were most useful, as shown below:

```
                        numPages++;
                }
                printk(KERN_INFO "getpages: 0x%lx present? = %d\n", curaddr, pre
sent);
        //if(!hasHugePages) {
                        pte_unmap(ptep); //because we have huge pages, we have f
ar much fewer pages, so we must unmap a.s.a.p.
        //}
                curaddr += PAGE_SIZE; //increment current address by the page si
ze, so we go to the next page...
        }

        return numPages;
}

//helper function
struct vm_area_struct *my_find_vma(struct mm_struct * mm, unsigned long address)
 {
        struct vm_area_struct * vma = mm->mmap; //mmap is a pointer to the first
 vma structure in the memory region
        while(vma) {
                printk(KERN_INFO "getpages: vma: 0x%lx ~ 0x%lx\n",
                                vma->vm_start, vma->vm_end);
                if((vma->vm_start <= address) &&
                                                                88,1-8        86%
```

Additionally, I also commented out code in getpages.c which I thought was not beneficial to the effiency of the code as shown below:

```
        //if(!hasHugePages) {
                        pte_unmap(ptep); //because we have huge pages, we have f
ar much fewer pages, so we must unmap a.s.a.p.
        //}
```

I also moved the system call code getpages.c to the linux kernel subdirectory ~/linux-2.6.11/kernel as so:

```
student@ubuntu:~/linux-2.6.11/kernel $ ls getpages.c
-rw-r--r--    1 student  student      2796 2021-04-30 08:47 getpages.c
student@ubuntu:~/linux-2.6.11/kernel $ _
```

Additionally I also added the system call number and ‘name’ manually, first by adding getpages.o to the kernel/Makefile as so:

```
student@ubuntu:~/linux-2.6.11/kernel $ ls Makefile
-rw-r--r--    1 student  student      2014 2021-04-30 05:05 Makefile
student@ubuntu:~/linux-2.6.11/kernel $ sudo vi Makefile
```

```
obj-y     = sched.o fork.o exec_domain.o panic.o printk.o profile.o \
            exit.o itimer.o time.o softirq.o resource.o \
            sysctl.o capability.o ptrace.o timer.o user.o \
            signal.o sys.o kmod.o workqueue.o pid.o \
            rcupdate.o intermodule.o extable.o params.o posix-timers.o \
            kthread.o wait.o kfifo.o sys_ni.o getproctimes.o getpages.o
```

Then, I made the function prototype in linux/include/syscalls.h, as shown below:

```
student@ubuntu:~/linux-2.6.11/include $ cd linux
student@ubuntu:~/linux-2.6.11/include/linux $ ls syscalls.h
-rw-r--r--    1 student  student     22859 2021-04-30 08:34 syscalls.h
student@ubuntu:~/linux-2.6.11/include/linux $ sudo vi syscalls.h
```

```
asmlinkage unsigned long getpages(unsigned long address);
```

Next, I added the new system call number and updated the max syscall definition include/asm-x86_64/unistd.h:

```
student@ubuntu:~/linux-2.6.11 $ sudo vi include/asm-x86_64/unistd.h_
```

```
#define __NR_getpages                252
__SYSCALL(__NR_getpages, sys_getpages)

#define __NR_syscall_max __NR_getpages
```

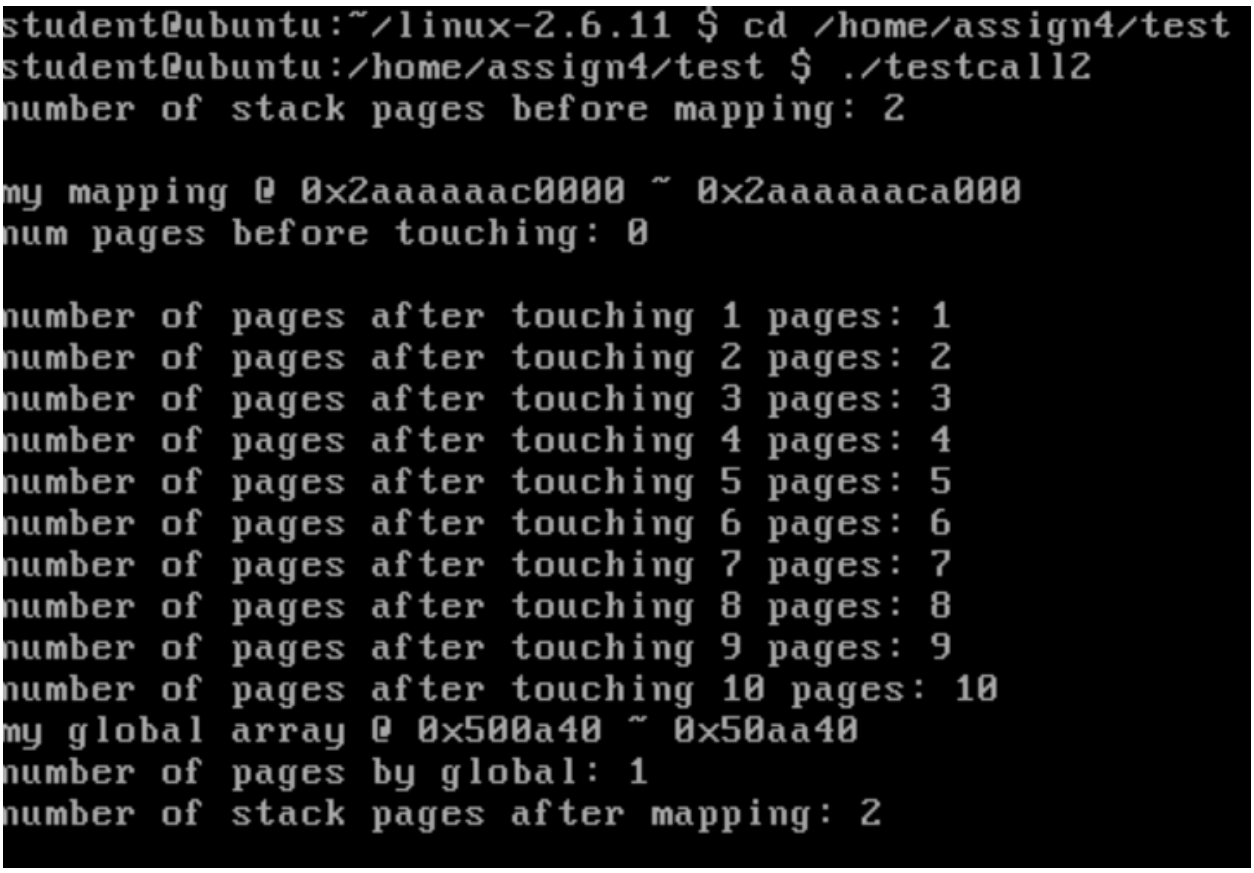
Then I wrote the userspace code testcall2.c under the /home/assign4/test directory as so, which is mostly verbatim from the video except that I also print the number of pages on the stack after the mapping is finished:

```
#include <stdio.h>

#include <stdlib.h>

#include <sys/mman.h>


//macro to define support of my getpages system call

#define getpages(arg1) syscall(252, arg1)


#define PAGE_SIZE 4096

#define NUM_MAP_PAGES 10

#define NUM_GLOB_PAGES 10

#define MY_GLOB_SIZE (NUM_GLOB_PAGES * PAGE_SIZE)

#define MY_MAP_SIZE (NUM_MAP_PAGES * PAGE_SIZE)
```

```
#define MY_MAP_PROT (PROT_READ | PROT_WRITE | PROT_EXEC)

#define MY_MAP_FLAGS (MAP_ANONYMOUS | MAP_PRIVATE)


char my_glob[MY_GLOB_SIZE] = "this is a test...";


int main(int argc, char *argv[]) {

        int i;

        unsigned long result = getpages(&result);

        printf("number of stack pages before mapping: %lu\n\n", result);


        void * my_mmap;

        void * my_mmap_ending;

        if((my_mmap = mmap(NULL, MY_MAP_SIZE, MY_MAP_PROT, MY_MAP_FLAGS, -1, 0)) == MAP_FAILED) {

                perror("mmap failed");

                return EXIT_FAILURE;

        }

        my_mmap_ending = (void *) ((char *)my_mmap + MY_MAP_SIZE);

        printf("my mapping @ %p ~ %p\n", my_mmap, my_mmap_ending);


        result = getpages(my_mmap);

        printf("num pages before touching: %lu\n\n", result);


        for(i = 0; i <NUM_MAP_PAGES; i++) {

                char * mod_mmap = (char *)(my_mmap + i * PAGE_SIZE);

                *mod_mmap = 'a';

                result = getpages(my_mmap);

                printf("number of pages after touching %d pages: %lu\n", (i + 1), result);

        }


        result = getpages(my_glob);

        char * my_glob_end = my_glob + MY_GLOB_SIZE; //creating a global variable

        printf("my global array @ %p ~ %p\n", &my_glob, my_glob_end); //global array is between my_glob and
my_glob_end...
```

```
	printf("number of pages by global: %lu\n", result);


	result = getpages(&result);
	printf("number of stack pages after mapping: %lu\n\n", result);
	//while(1); //busy wait for me to check /proc


	return EXIT_SUCCESS;
}
```

Thereafter I compiled and ran the code as so, and as shown below, it runs as intended, showing the number of stack pages before and after mapping, the mapping of the physical page in the memory region itself, the number of pages before and after 'touching', and the result of an array which starts at the calling of getpages with global and ending in a 'global end' which is the updated size of the global at this point as well as the size of it at this point.



Then I changed back to the ~/linux-2.6.11/ directory and did the system recompilation steps, sudo make, sudo make install, sudo make modules_install, sudo update grub, and finally sudo reboot:

```
student@ubuntu:~/linux-2.6.11 $ sudo make_
```

```
student@ubuntu:~/linux-2.6.11 $ sudo make install_
```

```
student@ubuntu:~/linux-2.6.11 $ sudo make modules_install
```

```
student@ubuntu:~/linux-2.6.11 $ sudo update-grub
```

```
student@ubuntu:~/linux-2.6.11 $ sudo reboot
```

Then finally I created the log directory, and created the log file dmesg.txt by directing the output of the circular buffer into it with the command ‘ dmesg -c > dmesg.txt’, and also creating a log file called ‘userspaceout.txt’, which is simply the output of the userspace code as so:

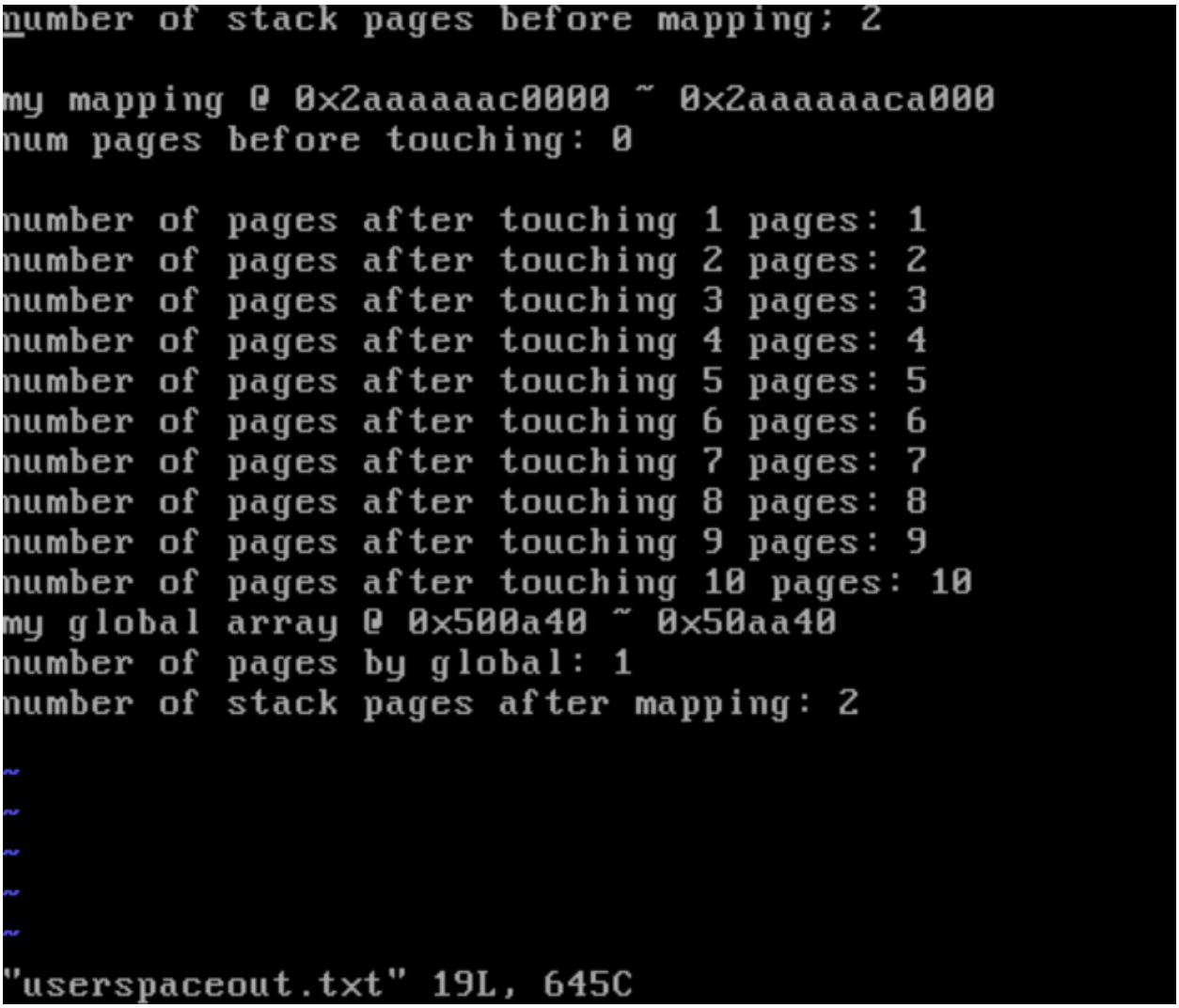
```
student@ubuntu:/home/assign4/test $ cd ..
student@ubuntu:/home/assign4 $ ls
total 20
drwxr-sr-x    4 student  student      4096 2021-05-05 23:23 .
drwxrwsr-x   10 student  student      4096 2021-04-29 13:24 ..
-rw-r--r--    1 root     student      2796 2021-05-05 23:23 getpages.c
drwxr-sr-x    2 root     student      4096 2021-05-05 21:29 log
drwxr-sr-x    2 student  student      4096 2021-05-05 21:31 test
student@ubuntu:/home/assign4 $ cd log
student@ubuntu:/home/assign4/log $ ls
total 28
drwxr-sr-x    2 root     student      4096 2021-05-05 21:29 .
drwxr-sr-x    4 student  student      4096 2021-05-05 23:23 ..
-rwxrwxrwx    1 root     student     15303 2021-05-05 21:28 dmesg.txt
-rwxrwxrwx    1 student  student       645 2021-04-30 09:59 userspaceout.txt
student@ubuntu:/home/assign4/log $ dmesg -c > dmesg.txt_
```

```
student@ubuntu:/home/assign4/log $ sudo vi dmesg.txt_
```

```
etpages: found vma containing 0x2aaaaaac0000: 0x2aaaaaac0000 ~ 0x2aaaaaaca000
getpages: 0x2aaaaaac0000 present? = 1
getpages: 0x2aaaaaac1000 present? = 1
getpages: 0x2aaaaaac2000 present? = 1
getpages: 0x2aaaaaac3000 present? = 1
getpages: 0x2aaaaaac4000 present? = 1
getpages: 0x2aaaaaac5000 present? = 1
getpages: 0x2aaaaaac6000 present? = 1
getpages: 0x2aaaaaac7000 present? = 0
getpages: 0x2aaaaaac8000 present? = 0
getpages: 0x2aaaaaac9000 present? = 0
getpages designed by Eric Samuel Huddleston is invoked
getpages invoked with address: 0x2aaaaaac0000
getpages: vma: 0x400000 ~ 0x401000
getpages: vma: 0x500000 ~ 0x50b000
getpages: vma: 0x2aaaaaaab000 ~ 0x2aaaaaabe000
getpages: vma: 0x2aaaaaabe000 ~ 0x2aaaaaac0000
getpages: vma: 0x2aaaaaac0000 ~ 0x2aaaaaaca000
getpages: found vma containing 0x2aaaaaac0000: 0x2aaaaaac0000 ~ 0x2aaaaaaca000
getpages: 0x2aaaaaac0000 present? = 1
getpages: 0x2aaaaaac1000 present? = 1
getpages: 0x2aaaaaac2000 present? = 1
getpages: 0x2aaaaaac3000 present? = 1
getpages: 0x2aaaaaac4000 present? = 1
```

```
student@ubuntu:/home/assign4/test $ ./testcall2 > userspaceout.txt_
```

```
student@ubuntu:/home/assign4/log $ sudo vi userspaceout.txt_
```


number of stack pages before mapping: 2
my mapping @ 0x2aaaaaac0000 ~ 0x2aaaaaaca000
num pages before touching: 0
number of pages after touching 1 pages: 1
number of pages after touching 2 pages: 2
number of pages after touching 3 pages: 3
number of pages after touching 4 pages: 4
number of pages after touching 5 pages: 5
number of pages after touching 6 pages: 6
number of pages after touching 7 pages: 7
number of pages after touching 8 pages: 8
number of pages after touching 9 pages: 9
number of pages after touching 10 pages: 10
my global array @ 0x500a40 ~ 0x50aa40
number of pages by global: 1
number of stack pages after mapping: 2
"userspaceout.txt" 19L, 645C