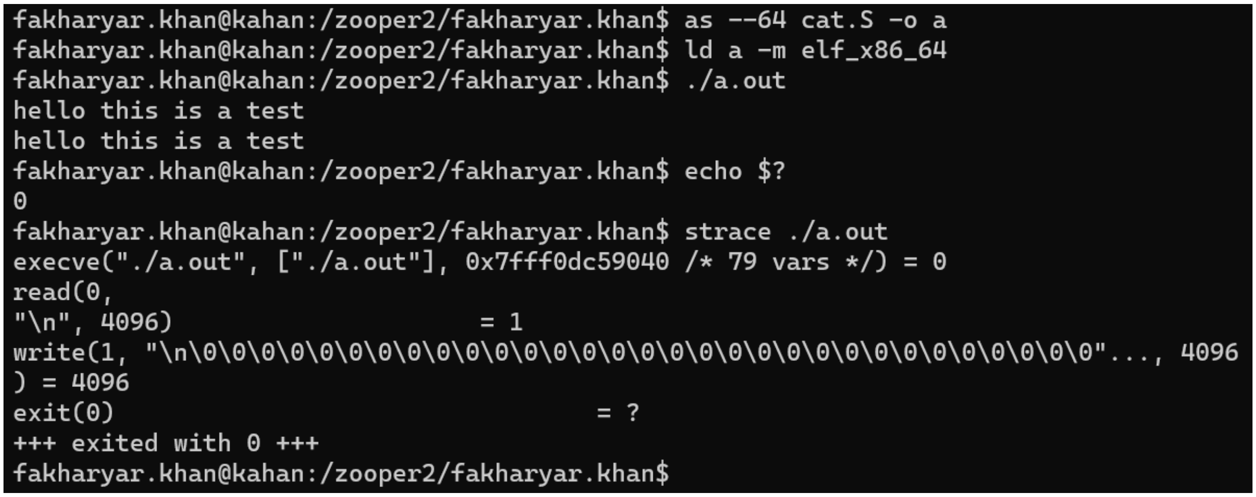
Hengnan Ma, Fakharyar Khan
December 14th, 2023
Operating Systems
Professor Hakner

Pset 7 Problem #2

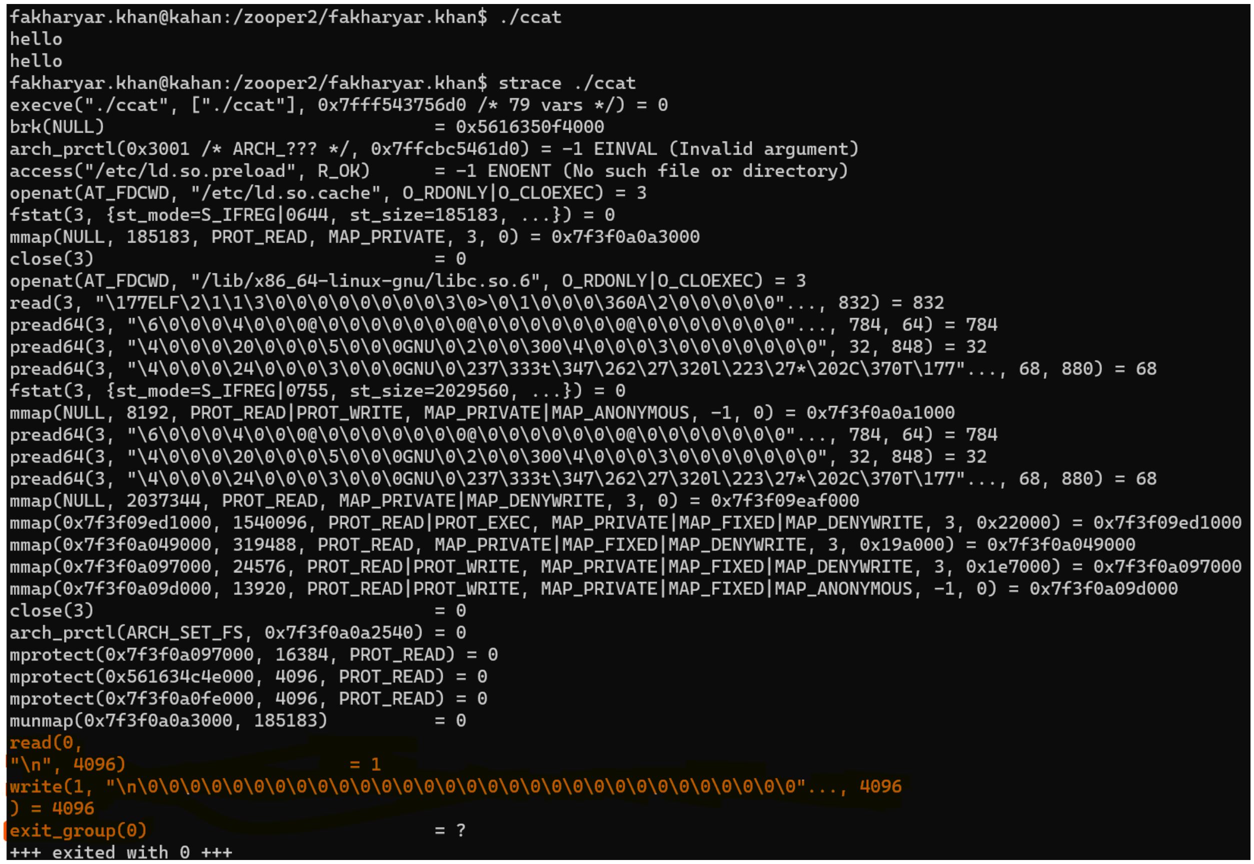
In this problem set, we created a simplified version of the cat program in x86-64 assembly. This program reads from standard input and writes back what was read to standard output. The program allocates 4096 bytes in the bss region for the buffer which will hold what was read from standard input. After calling on the read system call, we then write what was placed into the buffer into standard output and then call on the exit system call to exit the program with exit status 0.

As can be seen below, our program works as intended and returns an exit status of 0. And after running the strace command, we see that our program begins by calling on execve to run our executable. Then we call on the read, write, and exit system calls. One thing to note is that since our buffer is currently uninitialized, it's filled with null characters which is why the write system call has that long line of '\0's.



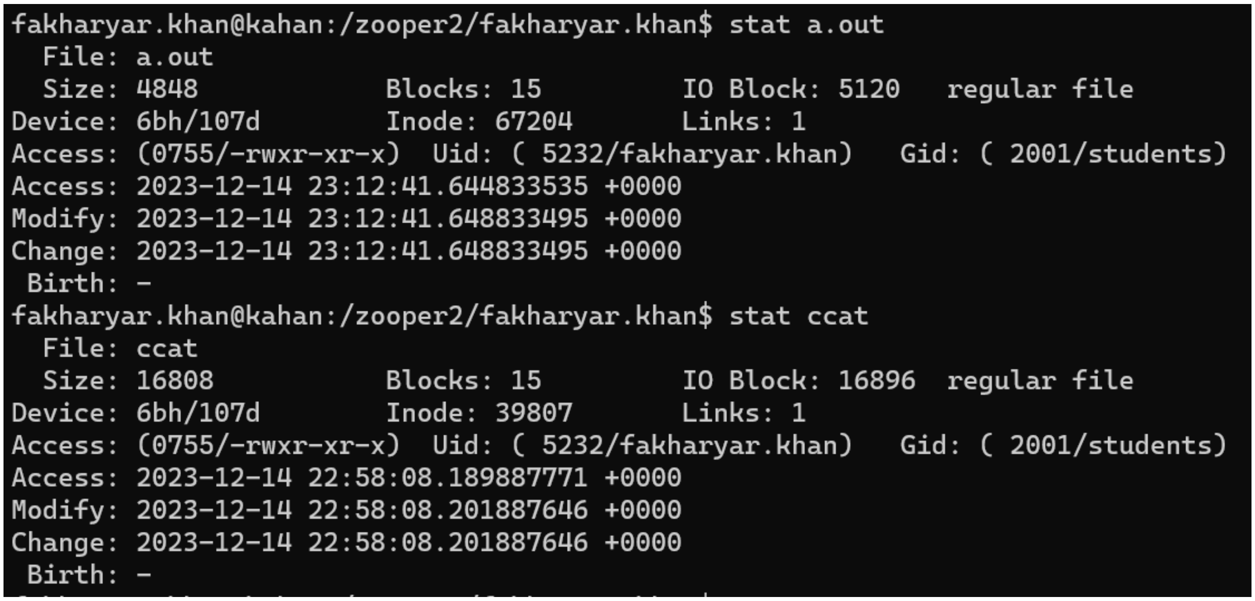
```
fakharyar.khan@kahan:/zooper2/fakharyar.khan$ as --64 cat.S -o a
fakharyar.khan@kahan:/zooper2/fakharyar.khan$ ld a -m elf_x86_64
fakharyar.khan@kahan:/zooper2/fakharyar.khan$ ./a.out
hello this is a test
hello this is a test
fakharyar.khan@kahan:/zooper2/fakharyar.khan$ echo $?
0
fakharyar.khan@kahan:/zooper2/fakharyar.khan$ strace ./a.out
execve("./a.out", ["./a.out"], 0x7fff0dc59040 /* 79 vars */) = 0
read(0,
"\n", 4096)                    = 1
write(1, "\n\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 4096
) = 4096
exit(0)                                 = ?
+++ exited with 0 +++
fakharyar.khan@kahan:/zooper2/fakharyar.khan$
```

On the other hand, when we write the equivalent code in C, it works exactly like the code written in assembly does. But, when we run the strace command, we see that there are many more system calls being made to perform the same function.



```
fakharyar.khan@kahan:/zooper2/fakharyar.khan$ ./ccat
hello
hello
fakharyar.khan@kahan:/zooper2/fakharyar.khan$ strace ./ccat
execve("./ccat", ["./ccat"], 0x7fff543756d0 /* 79 vars */) = 0
brk(NULL)                               = 0x5616350f4000
arch_prctl(0x3001 /* ARCH_??? */, 0x7ffcbc5461d0) = -1 EINVAL (Invalid argument)
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=185183, ...}) = 0
mmap(NULL, 185183, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f3f0a0a3000
close(3)                                = 0
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\360A\2\0\0\0\0\0"..., 832) = 832
pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784
pread64(3, "\4\0\0\0\20\0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0", 32, 848) = 32
pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\237\333t\347\262\27\320l\223\27*\202C\370T\177"..., 68, 880) = 68
fstat(3, {st_mode=S_IFREG|0755, st_size=2029560, ...}) = 0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f3f0a0a1000
pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784
pread64(3, "\4\0\0\0\20\0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0", 32, 848) = 32
pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\237\333t\347\262\27\320l\223\27*\202C\370T\177"..., 68, 880) = 68
mmap(NULL, 2037344, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f3f09eaf000
mmap(0x7f3f09ed1000, 1540096, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x22000) = 0x7f3f09ed1000
mmap(0x7f3f0a049000, 319488, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x19a000) = 0x7f3f0a049000
mmap(0x7f3f0a097000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1e7000) = 0x7f3f0a097000
mmap(0x7f3f0a09d000, 13920, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f3f0a09d000
close(3)                                = 0
arch_prctl(ARCH_SET_FS, 0x7f3f0a0a2540) = 0
mprotect(0x7f3f0a097000, 16384, PROT_READ) = 0
mprotect(0x561634c4e000, 4096, PROT_READ) = 0
mprotect(0x7f3f0a0fe000, 4096, PROT_READ) = 0
munmap(0x7f3f0a0a3000, 185183)          = 0
read(0,
"\n", 4096)                     = 1
write(1, "\n\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 4096
) = 4096
exit_group(0)                           = ?
+++ exited with 0 +++
```

Additionally, we see that the assembly executable file is less than a third of the size of the C executable file.



```
fakharyar.khan@kahan:/zooper2/fakharyar.khan$ stat a.out
  File: a.out
  Size: 4848            Blocks: 15         IO Block: 5120   regular file
Device: 6bh/107d        Inode: 67204       Links: 1
Access: (0755/-rwxr-xr-x)  Uid: ( 5232/fakharyar.khan)   Gid: ( 2001/students)
Access: 2023-12-14 23:12:41.644833535 +0000
Modify: 2023-12-14 23:12:41.648833495 +0000
Change: 2023-12-14 23:12:41.648833495 +0000
 Birth: -
fakharyar.khan@kahan:/zooper2/fakharyar.khan$ stat ccat
  File: ccat
  Size: 16808           Blocks: 15         IO Block: 16896  regular file
Device: 6bh/107d        Inode: 39807       Links: 1
Access: (0755/-rwxr-xr-x)  Uid: ( 5232/fakharyar.khan)   Gid: ( 2001/students)
Access: 2023-12-14 22:58:08.189887771 +0000
Modify: 2023-12-14 22:58:08.201887646 +0000
Change: 2023-12-14 22:58:08.201887646 +0000
 Birth: -
```