CS 342 - Section 1 15.02.2020
Munib Emre Sevilgen - 21602416

Homework 1 - Report

- I have installed the Ubuntu Desktop 64-bit 18.04 LTS on my hard-disk by deleting all partitions of the disk. Therefore, I had got a clean installation of Ubuntu instead of Windows operating system. The installation process was very easy and the installation package has handled everything including the disk partition, creation of the account, etc. The 10 Linux commands that I have learned are cd, ls, file, cp, mv, rm, mkdir, less, top and history.
- The vmlinuz-5.3.0-28-generic file is the kernel executable file and it resides at the /boot directory. The version of the kernel is 5.3.0-28-generic.
- The subdirectories at the root of the kernel source code are arch, block, certs, crypto, Documentation, drivers, fs, include, init, ipc, kernel, lib, LICENSES, mm, net, samples, scripts, security, sound, tools, usr, and virt.
- The syscall_64.tbl file is the kernel executable file and it resides at the /arch/x86/entry/syscalls directory in the root directory of the kernel source files. The system call names corresponding to system call numbers 5, 43, 123, and 220 are fstat, accept, setfsgid, and semtimedop respectively.
- The sample output of the “strace rm a.out example1.c example2.c” command:

```
execve("/bin/rm", ["rm", "a.out", "example1.c", "example2.c"], 0x7ffff1d61ef8 /* 62 vars */)
= 0
brk(NULL)                               = 0x55edb43d8000
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
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=97583, ...}) = 0
mmap(NULL, 97583, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7fca499c6000
close(3)                                = 0
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
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\260\34\2\0\0\0\0\0"..., 832) = 832
fstat(3, {st_mode=S_IFREG|0755, st_size=2030544, ...}) = 0
mmap(NULL,                    8192,                    PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7fca499c4000
mmap(NULL,                    4131552,                  PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7fca493c6000
mprotect(0x7fca495ad000, 2097152, PROT_NONE) = 0
mmap(0x7fca497ad000,          24576,                   PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1e7000) = 0x7fca497ad000
mmap(0x7fca497b3000,          15072,                   PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7fca497b3000
close(3)                                = 0
arch_prctl(ARCH_SET_FS, 0x7fca499c5540) = 0
```

```
mprotect(0x7fca497ad000, 16384, PROT_READ) = 0
mprotect(0x55edb380b000, 4096, PROT_READ) = 0
mprotect(0x7fca499de000, 4096, PROT_READ) = 0
munmap(0x7fca499c6000, 97583)           = 0
brk(NULL)                               = 0x55edb43d8000
brk(0x55edb43f9000)                     = 0x55edb43f9000
openat(AT_FDCWD, "/usr/lib/locale/locale-archive", O_RDONLY|O_CLOEXEC) = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=11731760, ...}) = 0
mmap(NULL, 11731760, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7fca48895000
close(3)                                = 0
ioctl(0, TCGETS, {B38400 opost isig icanon echo ...}) = 0
newfstatat(AT_FDCWD, "a.out", {st_mode=S_IFREG|0755, st_size=8328, ...}, AT_SYMLINK_NOFOLLOW) = 0
geteuid()                               = 1000
newfstatat(AT_FDCWD, "a.out", {st_mode=S_IFREG|0755, st_size=8328, ...}, AT_SYMLINK_NOFOLLOW) = 0
faccessat(AT_FDCWD, "a.out", W_OK)      = 0
unlinkat(AT_FDCWD, "a.out", 0)          = 0
newfstatat(AT_FDCWD, "example1.c", {st_mode=S_IFREG|0664, st_size=162, ...}, AT_SYMLINK_NOFOLLOW) = 0
newfstatat(AT_FDCWD, "example1.c", {st_mode=S_IFREG|0664, st_size=162, ...}, AT_SYMLINK_NOFOLLOW) = 0
faccessat(AT_FDCWD, "example1.c", W_OK) = 0
unlinkat(AT_FDCWD, "example1.c", 0)     = 0
newfstatat(AT_FDCWD, "example2.c", {st_mode=S_IFREG|0664, st_size=48, ...}, AT_SYMLINK_NOFOLLOW) = 0
newfstatat(AT_FDCWD, "example2.c", {st_mode=S_IFREG|0664, st_size=48, ...}, AT_SYMLINK_NOFOLLOW) = 0
faccessat(AT_FDCWD, "example2.c", W_OK) = 0
unlinkat(AT_FDCWD, "example2.c", 0)     = 0
lseek(0, 0, SEEK_CUR)                   = -1 ESPIPE (Illegal seek)
close(0)                                = 0
close(1)                                = 0
close(2)                                = 0
exit_group(0)                           = ?
+++ exited with 0 +++
```

- Real, user, and sys values are the real time, user CPU time, and system CPU time spent to execute a command respectively.
  cp: real 0m0,002s, user 0m0,002s, sys 0m0,000s
  mkdir: real 0m0,003s, user 0m0,003s, sys 0m0,000s
  ls: real 0m0,002s, user 0m0,002s, sys 0m0,000s
  rm: real 0m0,002s, user 0m0,002s, sys 0m0,001sx

- The output of the C program:

```
Time of the getpid: 0.000003 seconds
Time of the mkdir: 0.000050 seconds
Time of the open: 0.000003 seconds
Time of the write 100000 byte: 0.000067 seconds
Time of the write 10000 byte: 0.000014 seconds
Time of the write 1000 byte: 0.000011 seconds
Time of the write 100 byte: 0.000007 seconds
Time of the read 100000 byte: 0.000039 seconds
Time of the read 10000 byte: 0.000004 seconds
Time of the read 1000 byte: 0.000002 seconds
Time of the read 100 byte: 0.000001 seconds
```

The source code of the C program:

```
#include <stdio.h>
#include <sys/time.h>
#include <time.h>
#include <unistd.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <stdlib.h>

void timeval_subtract (struct timeval *result, struct timeval *left, struct timeval *right) {
  if (left->tv_usec < right->tv_usec) {
     int nsec = (right->tv_usec - left->tv_usec) / 1000000 + 1;
     right->tv_usec -= 1000000 * nsec;
     right->tv_sec += nsec;
  }
  if (left->tv_usec - right->tv_usec > 1000000) {
     int nsec = (left->tv_usec - right->tv_usec) / 1000000;
     right->tv_usec += 1000000 * nsec;
     right->tv_sec -= nsec;
  }

  result->tv_sec = left->tv_sec - right->tv_sec;
  result->tv_usec = left->tv_usec - right->tv_usec;
}

int main() {
  struct  timeval before;
  struct timeval after;
  struct timeval result;
```

```
    // getpid()
    gettimeofday(&before,NULL);
    getpid();
    gettimeofday(&after,NULL);
    timeval_subtract(&result, &after, &before);
    printf("Time of the getpid: %ld.%06ld seconds\n", result.tv_sec, result.tv_usec);

    // mkdir()
    rmdir("emre");
    gettimeofday(&before,NULL);
    mkdir("emre", 0777);
    gettimeofday(&after,NULL);
    timeval_subtract(&result, &after, &before);
    printf("Time of the mkdir: %ld.%06ld seconds\n", result.tv_sec, result.tv_usec);
    rmdir("emre");

    int fd;

    // open()
    gettimeofday(&before,NULL);
    fd = open("emre.txt", O_RDONLY, O_CREAT);
    gettimeofday(&after,NULL);
    timeval_subtract(&result, &after, &before);
    printf("Time of the open: %ld.%06ld seconds\n", result.tv_sec, result.tv_usec);
    close(fd);

    int bufferSize;

    // write() 100000 byte
    bufferSize = 100000;
    char buffer4[bufferSize];
    for (int i = 0; i < bufferSize; i++){
        buffer4[i] = rand()%256;
    }
    fd = open("100000byte", O_WRONLY | O_CREAT | O_TRUNC, 0644);
    gettimeofday(&before,NULL);
    write(fd, buffer4, bufferSize);
    gettimeofday(&after,NULL);
    timeval_subtract(&result, &after, &before);
        printf("Time of the write 100000 byte: %ld.%06ld seconds\n", result.tv_sec,
  result.tv_usec);
    close(fd);
    // write() 10000 byte
    bufferSize = 10000;
    char buffer3[bufferSize];
    for (int i = 0; i < bufferSize; i++){
       buffer3[i] = rand()%256;
    }
    fd = open("10000byte", O_WRONLY | O_CREAT | O_TRUNC, 0644);
    gettimeofday(&before,NULL);
    write(fd, buffer3, bufferSize);
    gettimeofday(&after,NULL);
    timeval_subtract(&result, &after, &before);
        printf("Time of the write 10000 byte: %ld.%06ld seconds\n", result.tv_sec,
  result.tv_usec);
    close(fd);

    // write() 1000 byte
    bufferSize = 1000;
    char buffer2[bufferSize];
    for (int i = 0; i < bufferSize; i++){
       buffer2[i] = rand()%256;
    }
    fd = open("1000byte", O_WRONLY | O_CREAT | O_TRUNC, 0644);
    gettimeofday(&before,NULL);
    write(fd, buffer2, bufferSize);
    gettimeofday(&after,NULL);
    timeval_subtract(&result, &after, &before);
        printf("Time of the write 1000 byte: %ld.%06ld seconds\n", result.tv_sec,
  result.tv_usec);
    close(fd);

    // write() 100 byte
    bufferSize = 100;
    char buffer1[bufferSize];
    for (int i = 0; i < bufferSize; i++){
       buffer1[i] = rand()%256;
    }
    fd = open("100byte", O_WRONLY | O_CREAT | O_TRUNC, 0644);
    gettimeofday(&before,NULL);
    write(fd, buffer1, bufferSize);
    gettimeofday(&after,NULL);
    timeval_subtract(&result, &after, &before);
    printf("Time of the write 100 byte: %ld.%06ld seconds\n", result.tv_sec, result.tv_usec);
    close(fd);
    // read() 100000 byte
    bufferSize = 100000;
    char buffer8[bufferSize];
    fd = open("100000byte", O_RDONLY, 0);
    gettimeofday(&before,NULL);
    read(fd, &buffer8, bufferSize);
    gettimeofday(&after,NULL);
    timeval_subtract(&result, &after, &before);
        printf("Time of the read 100000 byte: %ld.%06ld seconds\n", result.tv_sec,
  result.tv_usec);
    close(fd);

    // read() 10000 byte
    bufferSize = 10000;
    char buffer7[bufferSize];
    fd = open("10000byte", O_RDONLY, 0);
    gettimeofday(&before,NULL);
    read(fd, &buffer7, bufferSize);
    gettimeofday(&after,NULL);
    timeval_subtract(&result, &after, &before);
        printf("Time of the read 10000 byte: %ld.%06ld seconds\n", result.tv_sec,
  result.tv_usec);
    close(fd);

    // read() 1000 byte
    bufferSize = 1000;
    char buffer6[bufferSize];
    fd = open("1000byte", O_RDONLY, 0);
    gettimeofday(&before,NULL);
    read(fd, &buffer6, bufferSize);
    gettimeofday(&after,NULL);
    timeval_subtract(&result, &after, &before);
        printf("Time of the read 1000 byte: %ld.%06ld seconds\n", result.tv_sec,
  result.tv_usec);
    close(fd);

    // read() 100 byte
    bufferSize = 100;
    char buffer5[bufferSize];
    fd = open("100byte", O_RDONLY, 0);
    gettimeofday(&before,NULL);
    read(fd, &buffer5, bufferSize);
		gettimeofday(&after,NULL);
		timeval_subtract(&result, &after, &before);
		printf("Time of the read 100 byte: %ld.%06ld seconds\n", result.tv_sec, result.tv_usec);
		close(fd);

		remove("100byte");
		remove("1000byte");
  remove("10000byte");
  remove("100000byte");

  return 0;
}
```