HW 6

Mike Hanling

27 FEB 2017

Questions

1. (5 points) Why does the operating system perform system calls as oppose to having each user perform the same operations themselves?

   System calls can be used to allocate memory. If users were doing this themselves, then there is a high chance that important information will be overwritten due to negligence.

2. (10 points) Look up the following C functions in the man page, label them as either a system call or not a system call.

   (a) fread()

   NOT

   (b) write()

   SYSTEM CALL

   (c) stat()

   SYSTEM CALL

   (d) mmap()

   SYSTEM CALL

   (e) execv()

   NOT

3. (10 points) Run `ic221-up`. In the `hw/06/prob3` directory you'll find a compiled program. Use `ltrace` to enumerate the library function calls occurring under `main()`.

   3 strlen() puts() fflush()

4. (10 points) Run `ic221-up`. In the `hw/06/prob4` directory you'll find a compiled program. Use `strace` to enumerate the system function calls occurring under `main()`.

Could not get executable to run correctly. Supposed to run `strace ./trace-me-2 [arguments] > /dev/null` and then count all of the system calls that will be listed with exit codes/return values.

5. (20 points) Consider a file, `accts.dat`, which stores 1000 accounts formatted based on the defined structure. Using `open()` and `read()`, complete the program below to print them out:

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>

typedef struct{
  long acctnum;
  double bal;
  char acctname[1024];
} acct_t;

void read_accts(accts){
  //COMPLETE ME
}

int main(int argc, char *argv[]){
  acct_t accts[1000];

  read_accts(accts);

  int i;
  for(i=0;i<1000;i++){
    printf("%ld (%f) -- %s\n",
           accts[i].acctnum,
           accts[i].bal,
           accts[i].acctname);
  }

  close(fd);

}
```

```
void read_accts(accts){
  char filename[128];
  printf("File of accounts: ");
  scanf("%s", filename);

  fd = open(filename, O_RDONLY);

  for (int i = 0; i < 1000; i++) {
    read(fd, &(acct[i].acctnum), sizeof(long));
    read(fd, &(acct[i].bal), sizeof(double));
    read(fd, &(acct[i].acctname), 1024);
  }

  //not needed in main
  close(fd);
}
```

6. (10 points) Complete the following ORing options that matching the `fopen()` permissions:

   (a) `r`

   O_RDONLY

   (b) `w`

O_WRONLY |O_TRUNC |O_CREAT

(c) `a`

O_WRONLY |O_APPEND |O_CREAT

(d) `w+`

O_RDWR |O_CREAT |O_TRUNC

(e) `r+`

O_RDWR

7. (10 points) Consider a `umask` of 0750 (the leading 0 is to indicate a number written in octal). For the following `open()` permissions, what actual permission will the file get? You can write your answers in octal.

(a) 0777

0000

(b) 0640

0137

(c) 0740

0037

(d) 0501

0276

(e) 0651

0126

8. (5 points) Explain why the umask is considered a security feature.

It ensures that even when a system call to `open()` requests higher permissions than already set with `umask`, the OS will not allow for those high permissions to be set. It will default back the highest available from the `umask`.