

Advanced Unix Programming

Lab 7

Purva Tendulkar : 111403049

Q1. “The child “exec” call inherits the file descriptors of parent if Close_on_exec is not set”. Demonstrate with an example

Code :

(A) file ass1.c

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

int main() {
    char *buf[2];
    int pid, status;
    extern char **environ;
    int fd1 = open("test.txt", O_RDONLY);
    buf[0] = (char *)malloc(10);

    sprintf(buf[0], "%i", fd1);
    buf[1] = NULL;
    fcntl(fd1, F_SETFD, 0);
    printf("Close_on_exec not set\n");
    if ((pid = fork()) == 0)
        if (execve("/home/purva/Desktop/pracs/aup/ass7/a", buf, environ) == -1)
            exit(0);

    pid = wait(&status);
    lseek(fd1, 0, SEEK_SET);
    fcntl(fd1, F_SETFD, 1);
    printf("Close_on_exec set\n");
    if ((pid = fork()) == 0)
        if (execve("/home/purva/Desktop/pracs/aup/ass7/a", buf, environ) == -1)
            exit(0);

    pid = wait(&status);
    return 0;
}
```

(B) File a.c

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

int main(int argc, char *argv[], char *envp[]) {
    char buf[10];
    int x, i;
```

```

// Read contents from file
int fd = atoi(argv[0]);
x = read(fd, buf, 10);
if (x == -1) {
    printf("Read error\n");
    return 1;
}
for (i = 0; i < x; i++)
    printf("%c", buf[i]);
printf("\n");
return 0;
}

```

(C) File test.txt

abcdxyz

Input & Output Screenshots :

```

purva@purva-HP-Notebook: ~/Desktop/pracs/aup/ass7
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass7$ ls -l
total 36
-rwxrwxr-x 1 purva purva 8768 Oct  9 11:14 a
-rw-rw-r-- 1 purva purva 376 Oct  9 11:14 a.c
-rwxrwxr-x 1 purva purva 9189 Oct  9 11:14 ass1
-rw-rw-r-- 1 purva purva 712 Oct  9 11:14 ass1.c
-rw-rw-r-- 1 purva purva 7 Oct  9 10:53 test.txt
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass7$ gcc a.c -Wall -o a
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass7$ gcc ass1.c -Wall -o ass1
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass7$ ./ass1
Close_on_exec not set
abcdxyz
Close_on_exec set
Read error
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass7$

```

Q2. Write a program that takes a file name as an argument, opens the file, reads it and closes the file. The file should contain a string with the name of another application (e.g., 'ls' or 'ps' or any of your own applications) and the program forks a new process that executes the application named in the file.

Code :

```

#include <stdio.h>
#include <unistd.h>
#include <fcntl.h>
#include <stdlib.h>
#include <sys/wait.h>
#include <string.h>

```

```

#define NAME_MAX 50

int main() {
    char buf[NAME_MAX];
    int fd1 = open("text.txt", O_RDONLY);
    int c;
    if((c = read(fd1, buf, NAME_MAX)) < 0) {
        printf("read error\n");
        return 0;
    }
    if(buf[c - 1] == '\n')
        buf[c - 1] = '\0';
    else
        buf[c] = '\0';

    int pid;
    char *arg[2];
    arg[0] = buf;
    arg[1] = NULL;
    char cmd[NAME_MAX] = "./";
    strcat(cmd, buf);

    pid = fork();
    if (pid == 0) {
        if(execvp(cmd, arg) == -1) {
            printf("error\n");
            exit(0);
        }
    }

    return 0;
}

```

Input & Output Screenshots :

The screenshot shows a Linux desktop environment. In the background, a Sublime Text 2 editor is open with the C code from the previous block. In the foreground, a terminal window is open, showing the following commands and output:

```

purva@purva-HP-Notebook: ~/Desktop/pracs/aup/ass7/q2
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass7/q2$ cat text.txt
a
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass7/q2$ cat a.c
#include <stdio.h>
#include <unistd.h>
#include <fcntl.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {
    printf("Executing program a\n");
    return 0;
}
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass7/q2$ gcc a.c -o a
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass7/q2$ ./a
Executing program a
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass7/q2$ gcc ass2.c -Wall -o ass2
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass7/q2$ ./ass2 text.txt
Executing program a
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass7/q2$

```

Q3. Implement `cat < hw.txt > hw-copy.txt`

Code :

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

int main(int argc, char *argv[]) {
    if (argc != 3) {
        printf("Invalid number of arguments\n");
        exit(-1);
    }

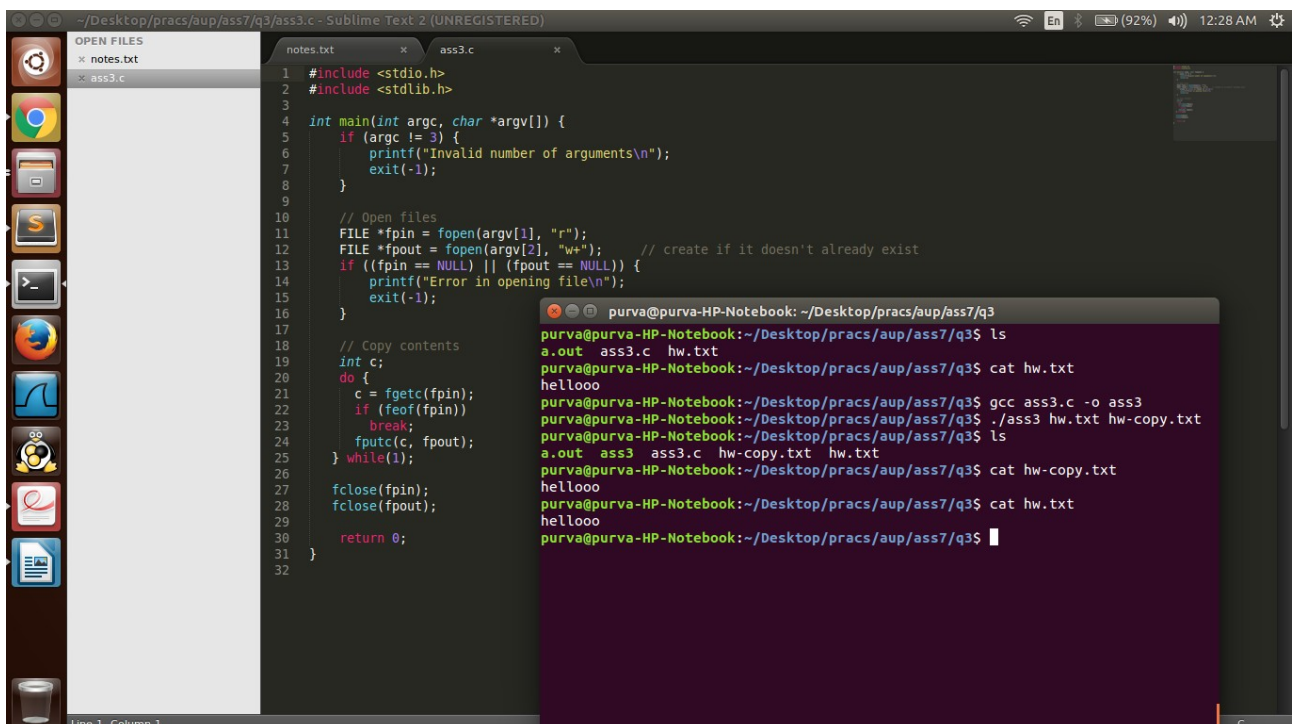
    // Open files
    FILE *fpin = fopen(argv[1], "r");
    FILE *fpout = fopen(argv[2], "w+"); // create if it doesn't already exist
    if ((fpin == NULL) || (fpout == NULL)) {
        printf("Error in opening file\n");
        exit(-1);
    }

    // Copy contents
    int c;
    do {
        c = fgetc(fpin);
        if (feof(fpin))
            break;
        fputc(c, fpout);
    } while(1);

    fclose(fpin);
    fclose(fpout);

    return 0;
}
```

Input & Output Screenshots :



```
~/Desktop/pracs/aup/ass7/q3/ass3.c - Sublime Text 2 (UNREGISTERED)
OPEN FILES
notes.txt
ass3.c
notes.txt
ass3.c
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main(int argc, char *argv[]) {
5     if (argc != 3) {
6         printf("Invalid number of arguments\n");
7         exit(-1);
8     }
9
10    // Open files
11    FILE *fpin = fopen(argv[1], "r");
12    FILE *fpout = fopen(argv[2], "w+"); // create if it doesn't already exist
13    if ((fpin == NULL) || (fpout == NULL)) {
14        printf("Error in opening file\n");
15        exit(-1);
16    }
17
18    // Copy contents
19    int c;
20    do {
21        c = fgetc(fpin);
22        if (feof(fpin))
23            break;
24        fputc(c, fpout);
25    } while(1);
26
27    fclose(fpin);
28    fclose(fpout);
29
30    return 0;
31 }
32

purva@purva-HP-Notebook: ~/Desktop/pracs/aup/ass7/q3
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass7/q3$ ls
a.out  ass3.c  hw.txt
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass7/q3$ cat hw.txt
hellooo
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass7/q3$ gcc ass3.c -o ass3
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass7/q3$ ./ass3 hw.txt hw-copy.txt
a.out  ass3  ass3.c  hw-copy.txt  hw.txt
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass7/q3$ cat hw-copy.txt
hellooo
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass7/q3$ cat hw.txt
hellooo
purva@purva-HP-Notebook:~/Desktop/pracs/aup/ass7/q3$
```

Q4. Bob works for an auditing agency needs to be able to read all the files in the system. The system admin has to protect the integrity of the system and should not allow Bob to modify or delete any file. Write a special SETUID program for the admin so that he can gave the executable permission of it to Bob. This program requires Bob to type a file name at the command line and then it will run /bin/cat to display the specified file. Can Bob compromise the integrity (by adding/modifying/deleting files) of this system? How?

Code :

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

int main(int argc, char *argv[]) {
    char *vec[3];
    if (argc < 2) {
        printf("Please type a file name.\n");
        return 1;
    }

    vec[0] = "/bin/cat"; vec[1] = argv[1]; vec[2] = 0;
    if (execve(vec[0], vec, 0) < 0) {
        printf("exec error\n");
        return 0;
    }

    return 0;
}
```

Contents of file temp.txt:

This is file temp.txt

Input & Output :

gcc ass4.c -Wall

./a.out temp.txt

This is file temp.txt

Explanation :

Bob can replace the "cat" executable in the bin directory and with his own malicious executable (with instructions to modify or delete files) with same name (i.e. cat). So when the program is run, it will execute the malicious cat file. Since the process has superuser privileges, the malicious cat program will also gain superuser privilege and can modify or delete any files on the system.