Assignment 2

Name: Jatin Kumar Sharma
Roll No: 2020563
Operastin System

Answer1:

The zip file contains 4 file, viz: main.c, E1.c, E2.c, Makefile

To run the program run command :
<Make> in the terminal opened in same directory where all files are stored.

LOGIC:

The main function first creates 3 child processes as described in the handout of the assignment2 viz: child_s1, child_sr, child_st.

Child S1: this register the signal handler using sigction syscall with flag siinnfo so that a signal can be queued.the handlr of signal first accepts the values in integer which in case of E1 prints directly but in case of E2 fuirst changed in to the required formt and then printed. The pid of this is passed to E1 and E2 using execlp syscall by converting the pid into string which then converted again to int using atoi glib function.

Child SR: First the setitimer is registered using the setitimer as taught in the class by our instructor, this register the Sigalrm handler using the sigaction syscall, the pid is converted to int datatype using atoi call, then function remain sleep using the whileloop so that it dont terminate in not working state.

The setitieme is set to 1s interval hence after each 1s SIGALRM interrupt generated to SR which is handeked using the SIGALRM handler and the handler of the function first read the random number using rdrand inline assemble and then queue it to the S1 process using sigqueue syscall and then a kill syscall been made with SIGTERM to S1 which is handed by S1's handler and this handler then reads this value that is stored in value parameter of siginfo struct, and it keeps repeating after every 1s as the interval is set for 1s.

Child ST: First the setitimer is registered using the setitimer as taught in the class by our instructor, this register the Sigalrm handler using the sigaction syscall, the pid is converted to int datatype using atoi call, then function remain sleep using the whileloop so that it dont terminate in not working state same as of SR.

The setitieme is set to 1s interval hence after each 1s SIGALRM interrupt generated to SR which is handeked using the SIGALRM handler and the handler of the function first read the cycle using rdtsc inline assemmblwhich then converted to format hhrrss and passed to the union strict int and then queued it to the S1 process using sigqueue syscall and then a kill syscall been made with SIGTERM to S1 which is handed by S1's handler and this handler then reads this value that is stored in value parameter of siginfo struct, and it keeps repeating after every 1s as the interval is set for 1s.

Answer2:

The zip file contains a folder answer2 which contains file, viz: ans2.c, assmt2q2.patch.

Assmt2q2.patch: this is the diff file between the stock and modified kernel in patch format as said in the handout.

Syscall: Firstly we make a syscall, which accepts 3 arguments viz: void pointer for destination file, void pointer for source file and no of bytes to be copied, for this we have defined a buff of length 256, the content of the src file then copied to this buff of the provided size using __copy_from_user syscall of kernel level and thenagan copied to user level variable dest using __copy_to_user.

The dest and src are pointing to user-level memory location hence can't communicate with the kernel directly so we first copy the content of src into the kernel level buff the kernel level buff to user level dest and hence the operation been done. Before doing the syscall we check for the validation of src and dest memory location using access_ok syscall which verify them to readable and writable.

The syscall is then listed to syscall_64/tbl and then we ran make command for compilation of the modified kernel then make modules_install followed by some more commands as tought in tutoria and refreseer module. In this we have our kernel with our syscall added we have made a program to test is which is as follows.

Ans2.c:
--> This is a c program which helps us to test the working of our syscall and for checking whether the syscall works perfectly we also print error if not works fine.This will first show the floating matrices before running syscall and after running syscall. So that it can be checked whether it shows correct output or not. The matrix is hardcode as source matrix and then a memory la=ocatin is generated using malloc function where the matrix will be copied.

The syscall been made as the hardcooded matrix for src, created location as dest, size as size of src matrix the syscall produced the desired output.

-Thank and Regards