Jatin Kumar Sharma

Roll No: 2020563

Operating System

Assignment-5

There are 2 programs viz kernel_c and kernel_p.

Kerenl_p: The program uses a function named as readData to reads 8 random bytes from /dev/urandom using a file descriptor and stores it into a pointer assigned a space of 8 bytes using malloc and passed to it. Then this data is handed over to the kernel syscall named writer(explained in the writer section of the readme). Now it is printed in this program to say that this data has been produced. if the queue is full then it prints that the queue is full.

Kerenl_c: The program passes a pointer have assigned a space of 8 bytes using malloc to a kernel syscall named reader, which consumes the data stored in kernel queue and simply print it to the console, if the queue is empty it prints that the queue is empty.

Moreover, there are 2 syscalls being programmed in sys.c which are then listed to compiled in syscalls_64.tbl. The kernel used for the program is globally defined in the file sys,c which is implemented using LinkedList having constant size of 3. The struct of the node of the queue contains 3 parameters as follows;

```
        struct qnode {
                char data[8];
                struct qnode *next;
                struct qnode *prev;
        };
```

The queue has two parameters as head and tail globally defined and a declared kernel leve semaphore to make sure the mutual exclusion around the critical section.

Writer syscall: 449: The data passes from kernel_p to this syscall first been copied to kernel space using __copy_from_user along with the access check then it is added to simply to the queue by storing it to the new qnode and connecting it to the queue the part when the node been attached to the queue is defined as critical section hence down_write(&queue_sem); been used to make sure about mutual exclusion and after adding the data again up_write(&queue_sem); been used to free the critical section and thus the data been added to the kernel queue.

reader syscall: 450: The data space passes to kernel_c to this syscall first been checked for access ok. After which the data present at the head of queue is copied to a local char pointer location using memcpy and the same node beeing removed fom the queue. then this data being copied to user level data space passed to this syscall using __copy_to_user. The part when the node been detached from the queue is defined as critical section hence down_read(&queue_sem); been used to make sure about mutual exclusion and after deleting the node again up_read(&queue_sem); been used to free the critical section and thus the data been added to the kernel queue.