Assignment 2_2

Overall Methodology

1. Write a syscall to do the task (syscall implementation explained later)
2. Add the syscall in the syscall table as well as the kernel sys.c file
3. Run make again with the subsequent kernel compilation steps to get the new syscall available for use
4. Extracted 2 fresh copies of the kernel and modified one with the changes by copying the updated sys.c and syscall_64.tbl into one extracted copy and created the patch using git diff command

syscall

```
SYSCALL_DEFINE4(kernel_2d_memcpy, float *, dst, float *, src, int, row_count, int,
col_count)
{
    float temp[row_count][col_count];
    int op1Failed = __copy_from_user(temp, src, sizeof(temp));
    if (op1Failed)
    {
        printk("copy_from_user failed to copy: look at internal implementation
under SYSCALL_DEFINE4 for more details\n");
        return -1;
    }
    int op2Failed = __copy_to_user(dst, temp, sizeof(temp));
    if (op2Failed)
    {
        printk("copy_to_user failed to copy: look at internal implementation under
SYSCALL_DEFINE4 for more details\n");
        return -1;
    }
    return 0;
}
```

1. The syscall takes in the destination, source and size of the matrix by taking in rows and columns
2. It then creates a temporary matrix which acts as an intermediate matrix during copying
3. Using the __copy_from_user() function the source matrix is copied into temp matrix. It's 3 parameters are destination, source and size of memory being transferred which can be conveniently set to sizeof(temp) here however an alternative approach could be to set it to row_count*column_count*sizeof(float). The syscall return 0 if it's successful else it returns a number greater than 0 in which case we use printk() function to add this print command to the kernel log and return -1.
4. Similarly the __copy_to_user() function then takes the temp matrix and copies it to the destination using similar logic as above by allocating a space equal to sizeof(temp). The syscall return 0 if it's successful else it returns a number greater than 0 in which case we use printk() function to add this print command to the kernel log and return -1.
5. Finally if the syscall executes the actions so far it returns 0 and hence the copying is successful.

Patch File

The enclosed patch file patchfile.patch is generated by using git diff against a default and a modified kernel. It reflects the changes in the linux-5.14.4/arch/x86/entry/syscalls/syscall_64.tbl and linux-5.14.4/kernel/sys.c files

Program to Show Working -

The program is present in the test.c file, the commands to run it are in the Makefile so simply run make after setting up the kernel and the file will run.

Alternatively the commands to run the test.c file without the make file are -

```
gcc test.c
./a.out
```

The program will then prompt to either use a default input or a custom input and the command line based instructions will be self-explanatory and print out the post copying result