COMPUTER NETWORKS LAB SECTION - 1

Lab Assignment-4

Harshiv Patel 1741005
Shyam Patel 1741030

1. **Objective**

   To implement sliding window ARQ to provide reliability in file transfer with UDP. As we saw packets being lost when sending at high data rates.

2. **Problems faced/Functions used**

1. **Displaying complete list of files**

int scandir(DirectoryName,NameList,Select,Compare)

This function scans the current directory and updates the input file with a complete list of * files and directories present in the current folder.

Problems:

We were trying to store the list of files in a char** variable. For that we used scandir() function. scandir() finished properly but we always got a segmentation fault when trying to print the char**. It was solved when we came to know that function scandir() allocates the memory for us. We never needed to allocate ANY memory. Although, we still needed to free up the memory.

2. **Reusing the received buffer**

int setsockopt(int *s*, int *level*, int *optname*, const void **optval*, socklen_t *optlen*);

For a server application if it crashes, then we don't want to wait a certain number of minutes before the kernel let us reuse the port avoiding the "Address already in use"

error messages. This can be avoided if you use the SO_REUSEADDR option, letting other sockets to bind to the same port unless there is an active listener bound already.

3. For clearing the buffer

The C library function **void *memset(void *str, int c, size_t n)** copies the character **c** (an unsigned char) to the first **n** characters of the string pointed to by the argument **str**.
Above function is used to clear all the data buffer and structure
E.g. memset(ack_send, 0, sizeof(ack_send));

The client side implements a user interaction module and publishes the supported commands. The following section covers in detail on how client and server interact on each user command.

The supported user command includes,

get [filename] - This command is used to get the specified file from server to client
==========

3. Implementation

SERVER SIDE

1.) On receiving the 'get' request from client, the server first checks if the filename is specified or not. If filename is null, then server does not send anything back to client
2.) Then it checks if the specified file is present and have appropriate read permission. If the file is not present or does not have appropriate permission, then the server does not send anything back to the client
3.) If the specified file is valid, then it sets a timeout of 2 seconds for the receive call on the server socket
4.) After this it opens the specified file, gets the file size and calculates the number of frames required to send the file
5.) First it sends the total number of frames and then checks if the received acknowledgement matches to the total number of frames

6.) Finally it sends all the frames sequentially and check the received ack. If ack does not match then it keeps resending the frames till the ack matches

CLIENT SIDE

1.) Server does not transfer any message if the filename is NULL or not present in the directory
2.) So, it sets a timeout of 2 seconds for the receive call on the client packet. If client does not receive the number of frames in 2 seconds, timeout will occur
3.) Disable the timeout after receiving the total number of frames
4.) Receive all the frames sequentially and then send frame ID as the Ack
5.) Discard the duplicate frames.
6.) Write the received data frame into a file.

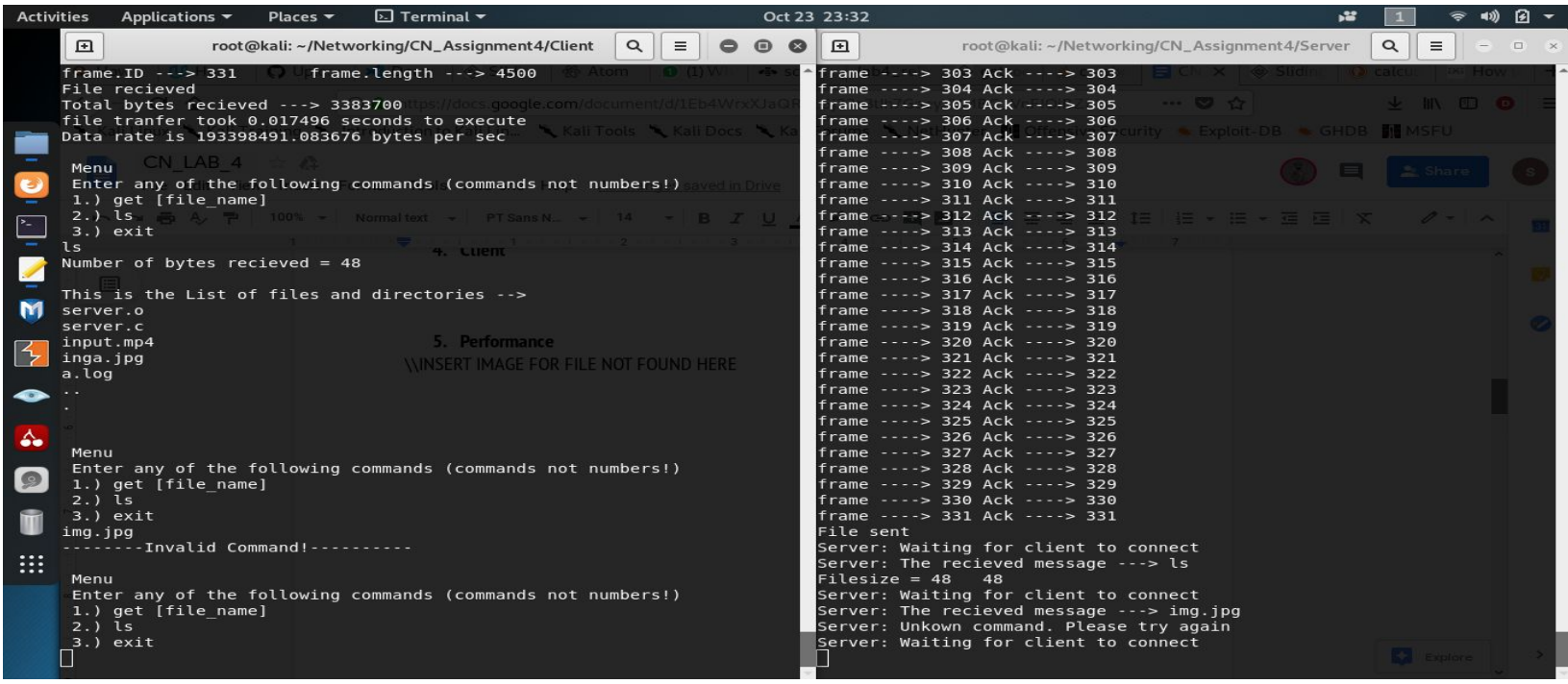
Other options:

ls - list all the files present in the server side

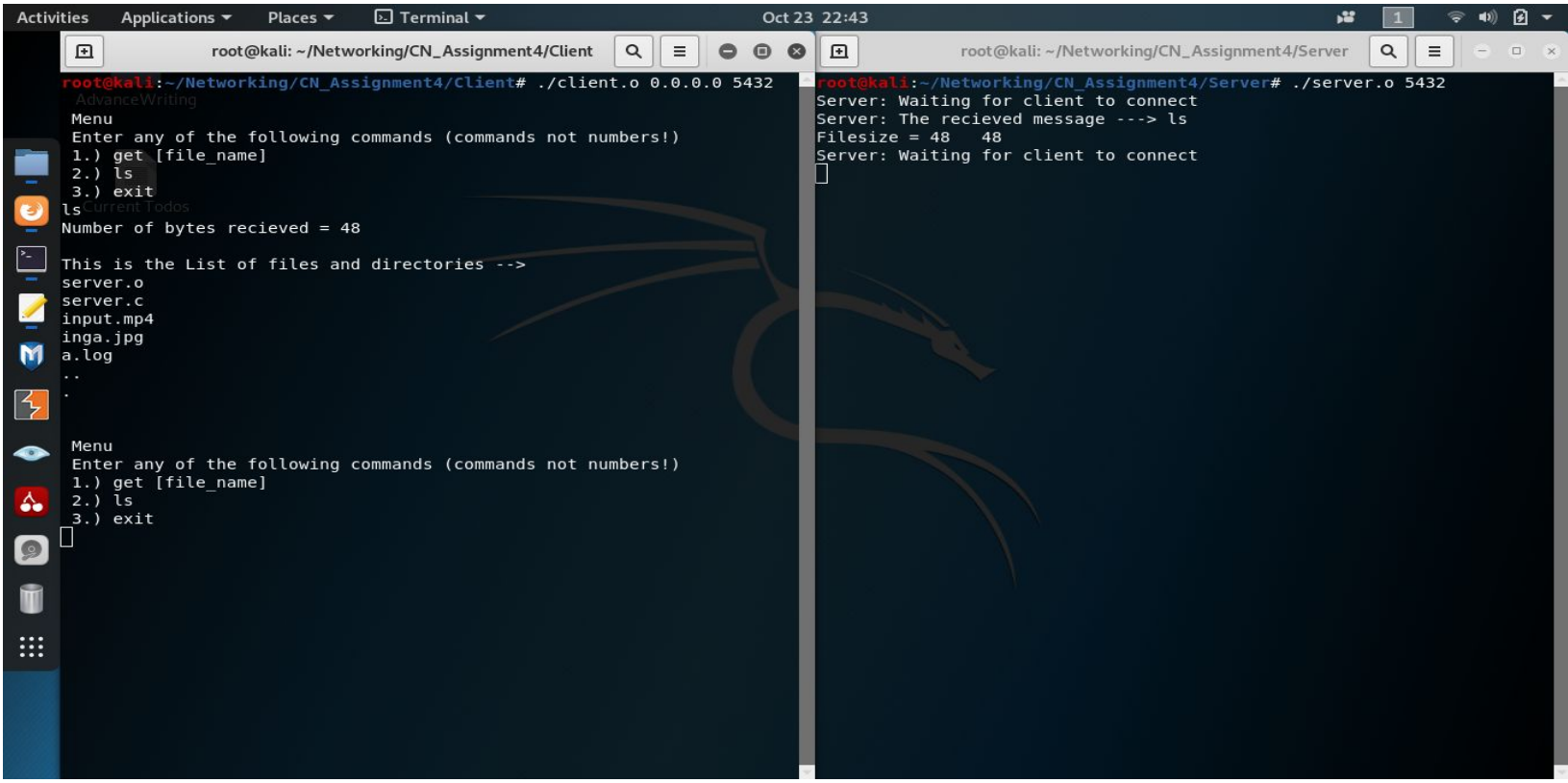
In this case, the server scans all the files and directories in the present folder and transfers
the entire list to the client.

exit - Close the client and server

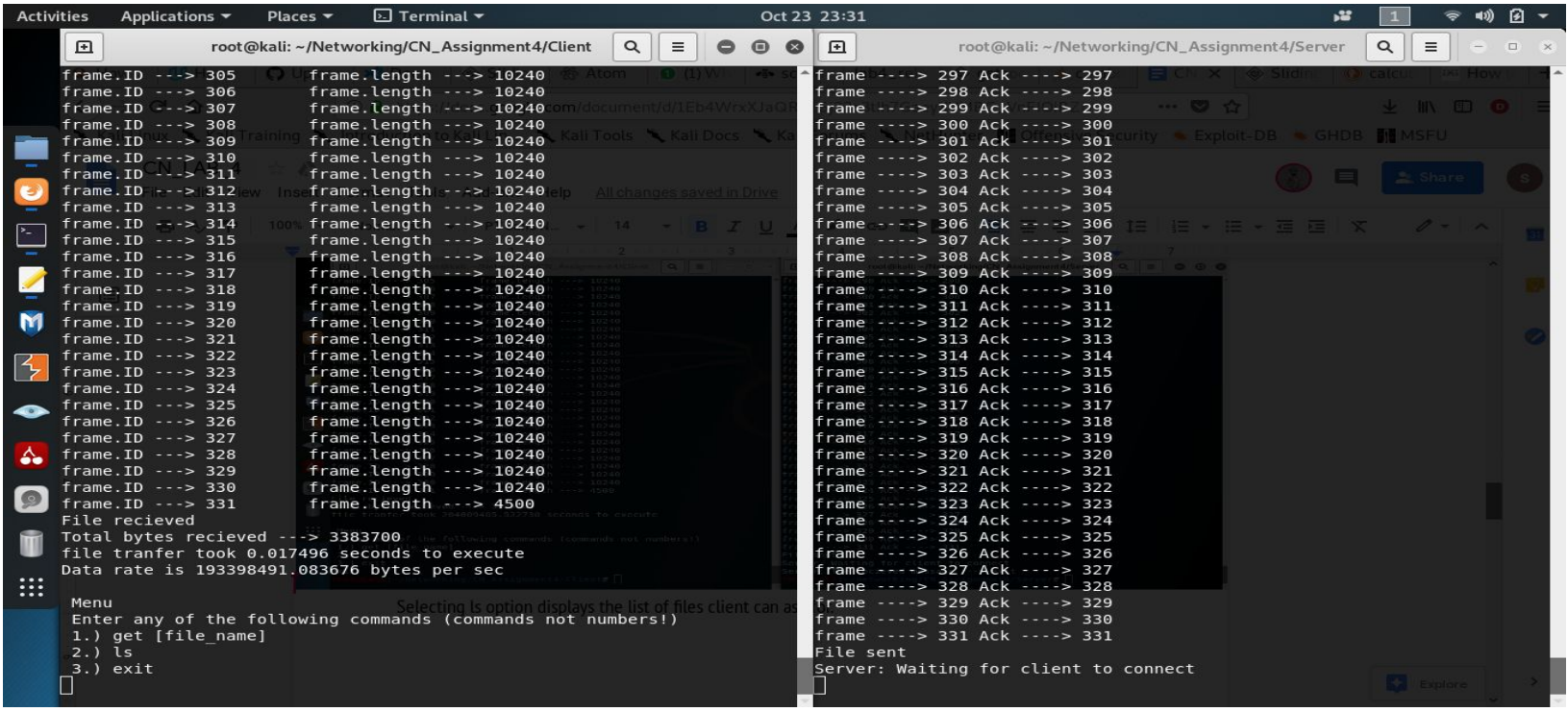
4. Performance



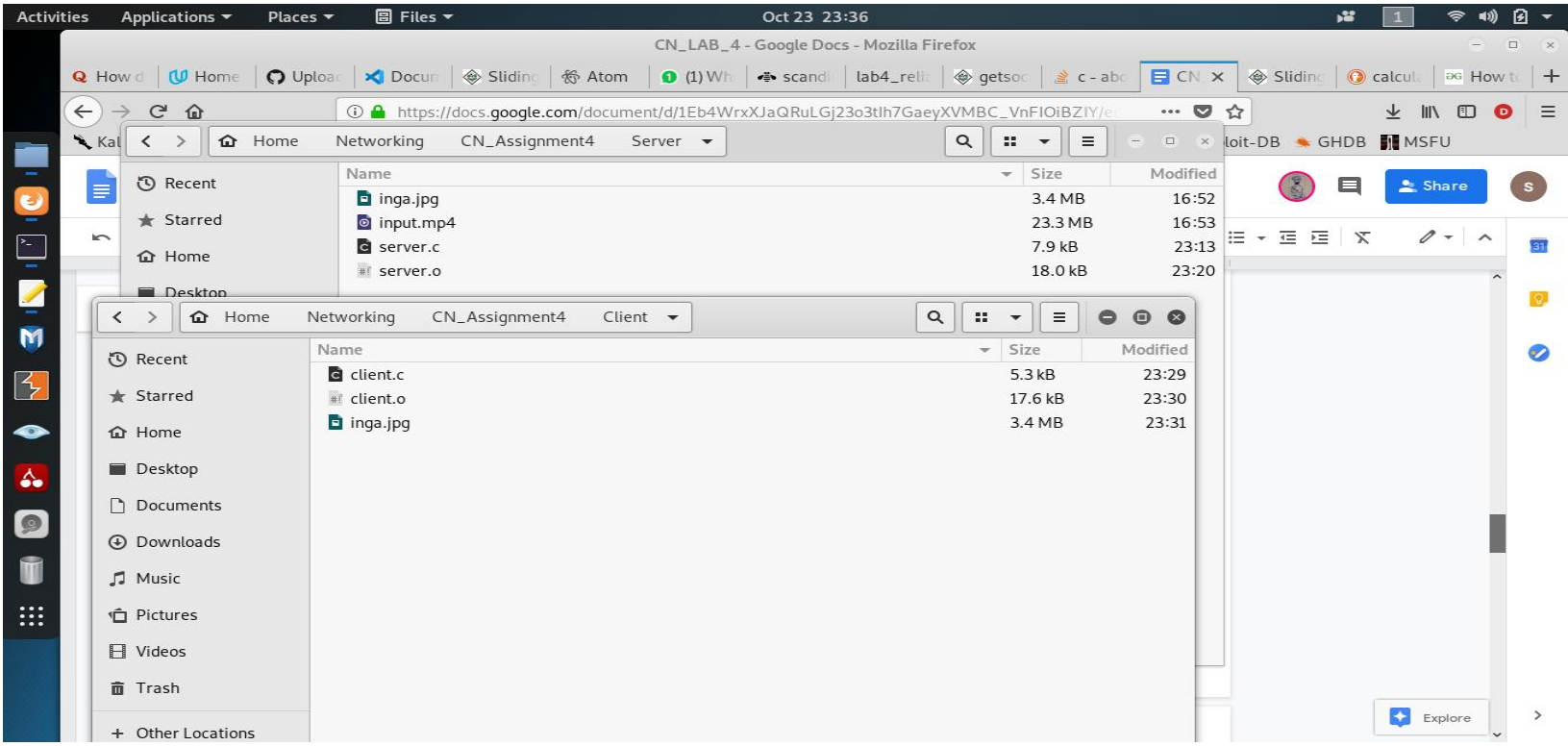
When the server does not have file it displays that the operation is invalid.



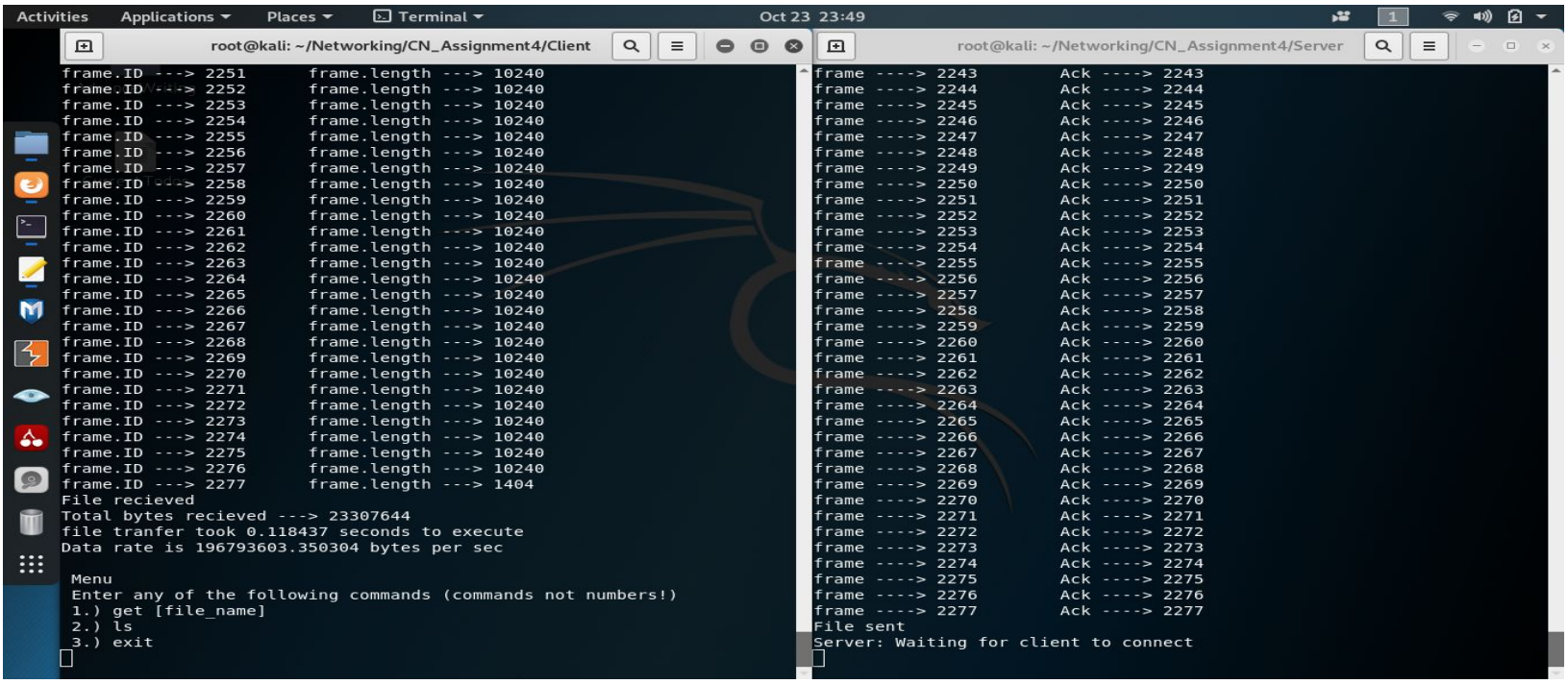
Selecting ls option displays the list of files client can ask for.

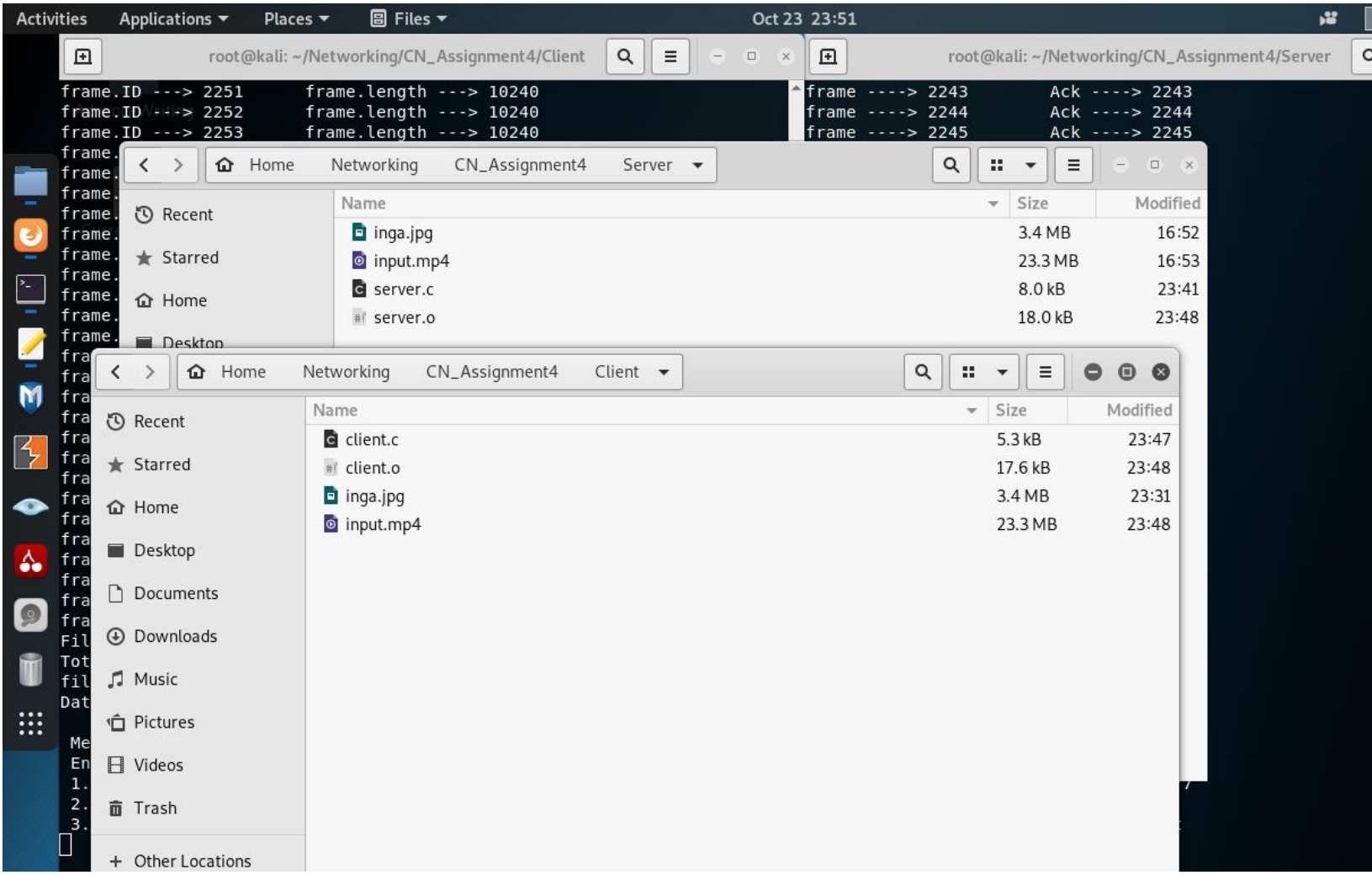


Buffer size is 10240 bytes. Here, SWS(=RWS) is of 4 character bytes(i.e 32 bytes). As, inga.jpg exists with the server we start receiving packets.



Transferring video of 10 MB.




Activities
Applications
Places
Files
Oct 23 23:51
root@kali: ~/Networking/CN_Assignment4/Client
root@kali: ~/Networking/CN_Assignment4/Server
frame.ID ---> 2251 frame.length ---> 10240
frame.ID ---> 2252 frame.length ---> 10240
frame.ID ---> 2253 frame.length ---> 10240
frame ----> 2243 Ack ----> 2243
frame ----> 2244 Ack ----> 2244
frame ----> 2245 Ack ----> 2245
Home
Networking
CN_Assignment4
Server
Recent
Starred
Home
Desktop
Name
Size
Modified
inga.jpg 3.4 MB 16:52
input.mp4 23.3 MB 16:53
server.c 8.0 kB 23:41
server.o 18.0 kB 23:48
Home
Networking
CN_Assignment4
Client
Recent
Starred
Home
Desktop
Documents
Downloads
Music
Pictures
Videos
Trash
Other Locations
Name
Size
Modified
client.c 5.3 kB 23:47
client.o 17.6 kB 23:48
inga.jpg 3.4 MB 23:31
input.mp4 23.3 MB 23:48