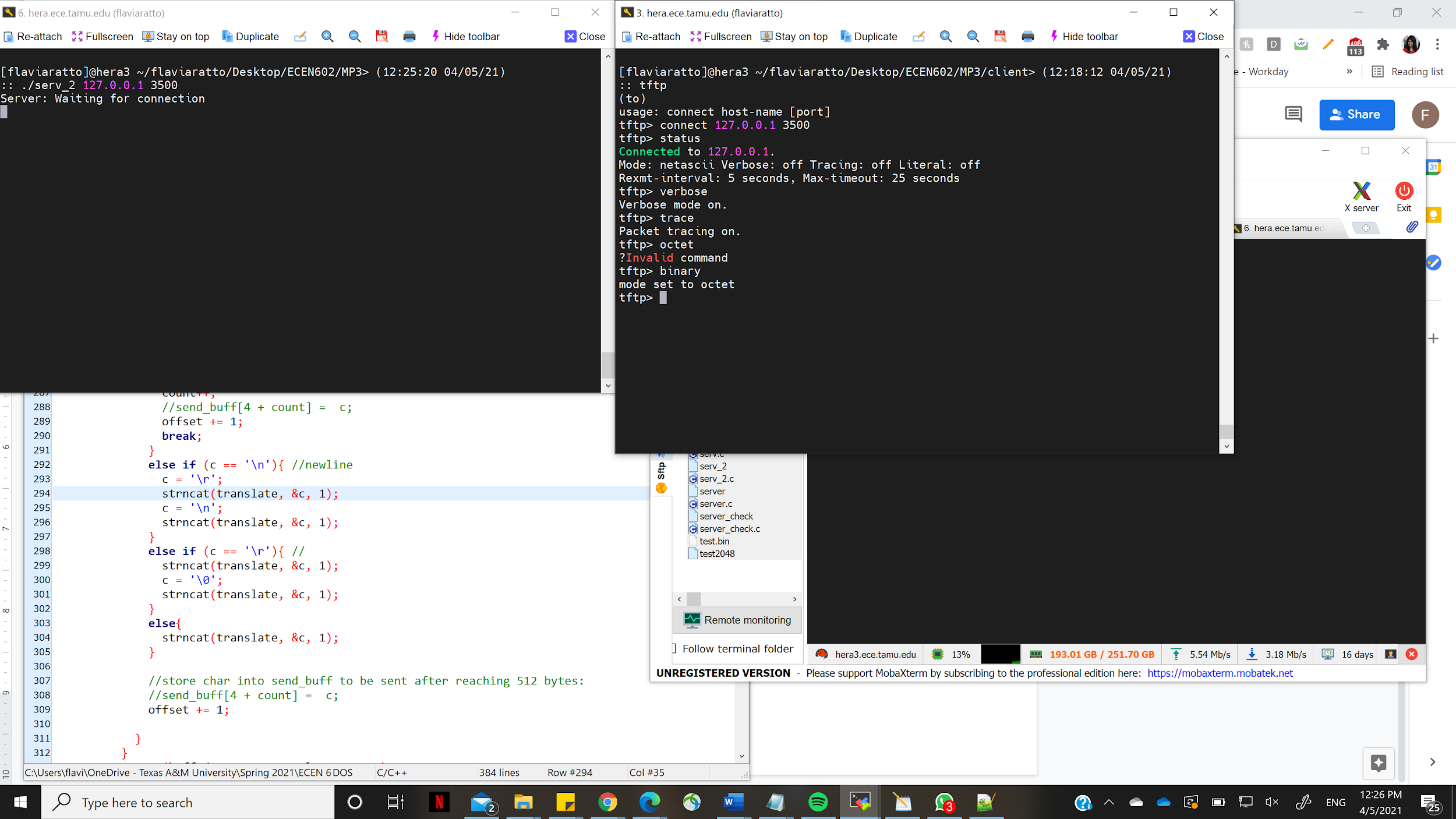
**Experimentation and Results**

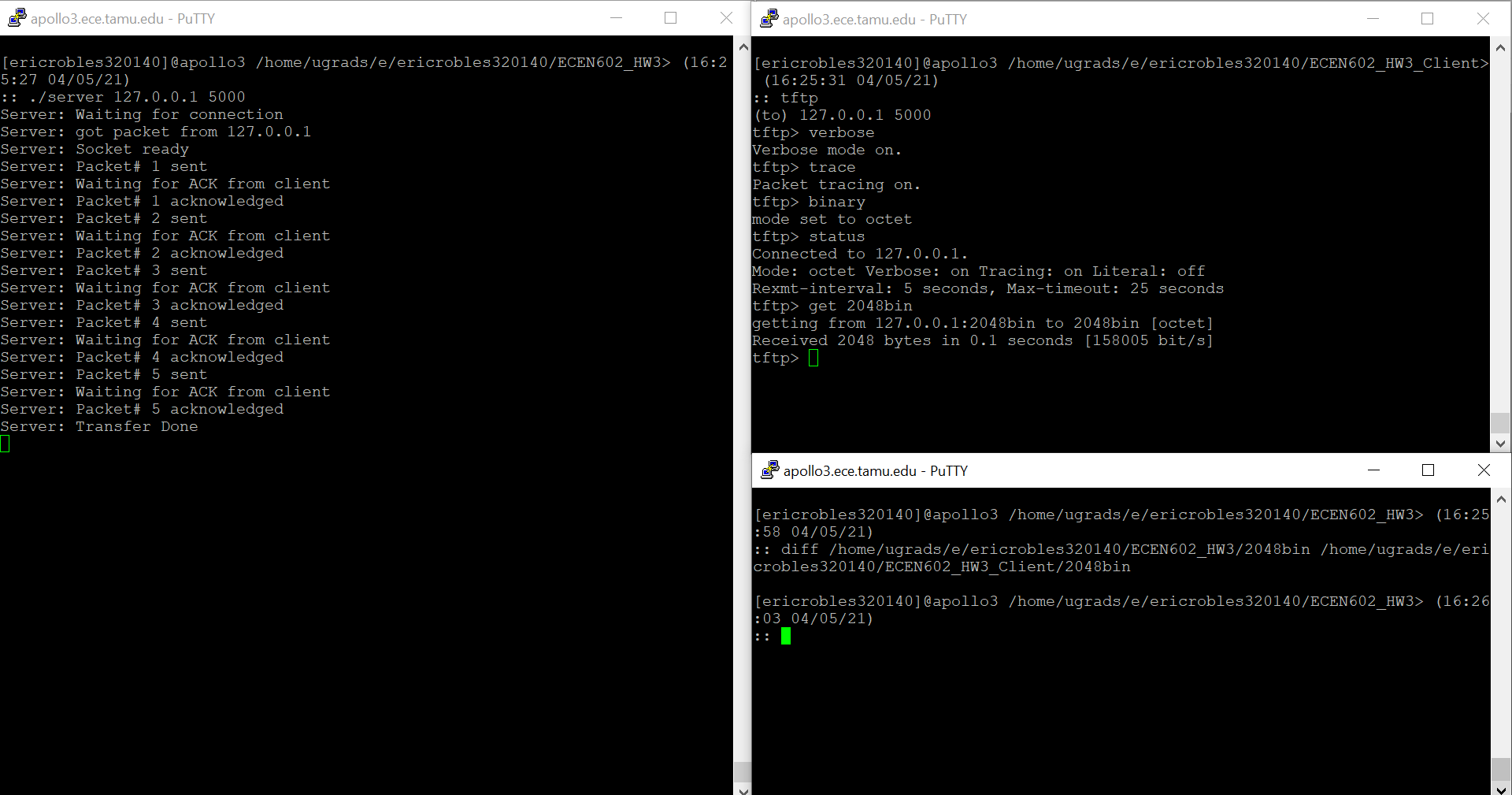
**Initial setup of client and server**

****

We have tested our program for the following test cases -

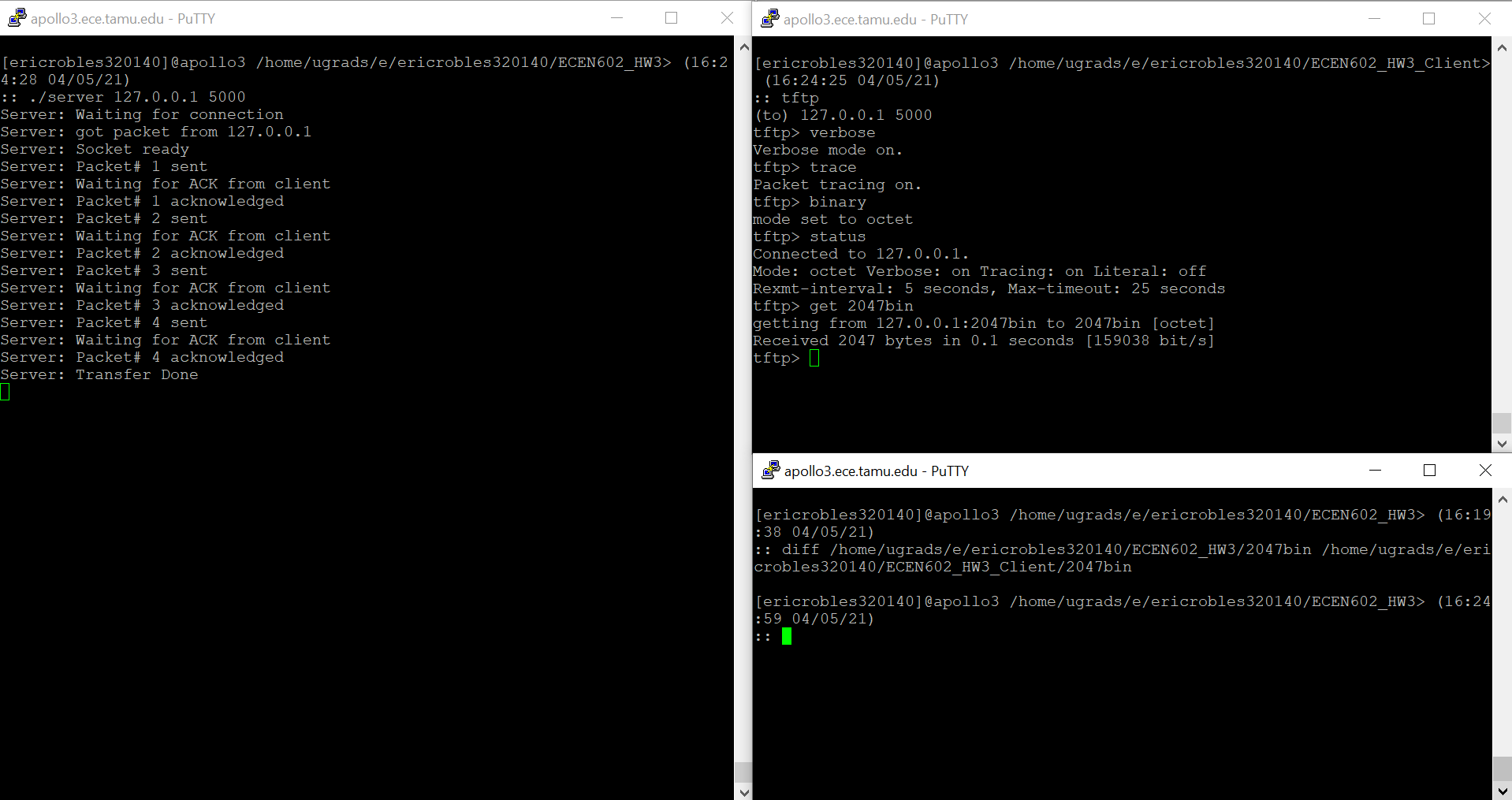
1. **Transfer a binary file of 2048 bytes and check that it matches the source file**

We used ECEN602\_HW3 as the local directory where server executable file and test files are located and where the server is launched. We used ECEN602\_HW3\_Client as a directory where tftp client is launched and retrieves the following file. We used diff in a separate terminal located on bottom right corner to check if the two files from two directories matches:



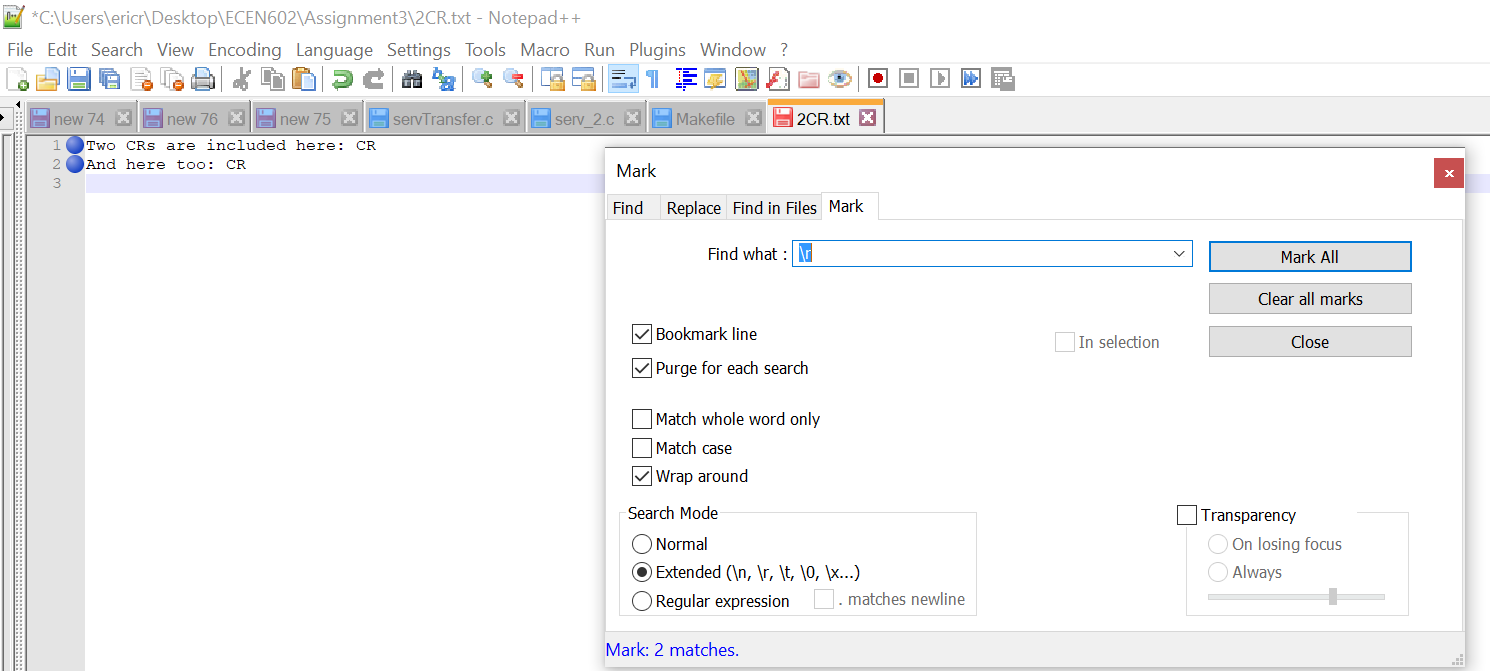
1. **Transfer a binary file of 2047 bytes and check that it matches the source file**

We used ECEN602\_HW3 as the local directory where server executable file and test files are located and where the server is launched. We used ECEN602\_HW3\_Client as a directory where tftp client is launched and retrieves the following file. We used diff in a separate terminal located on bottom right corner to check if the two files from two directories matches:



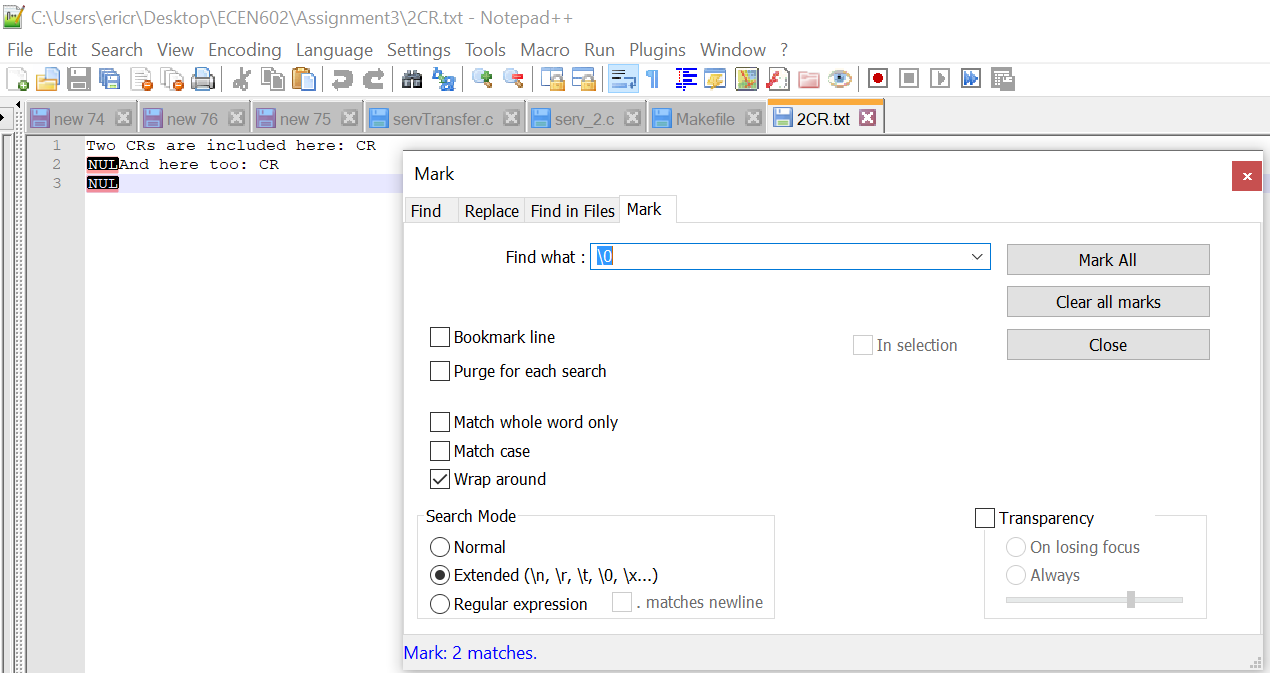
1. **Transfer a netascii file that includes two CR’s and check that the resulting file matches the input file,**

Test Text File with Two CRs:



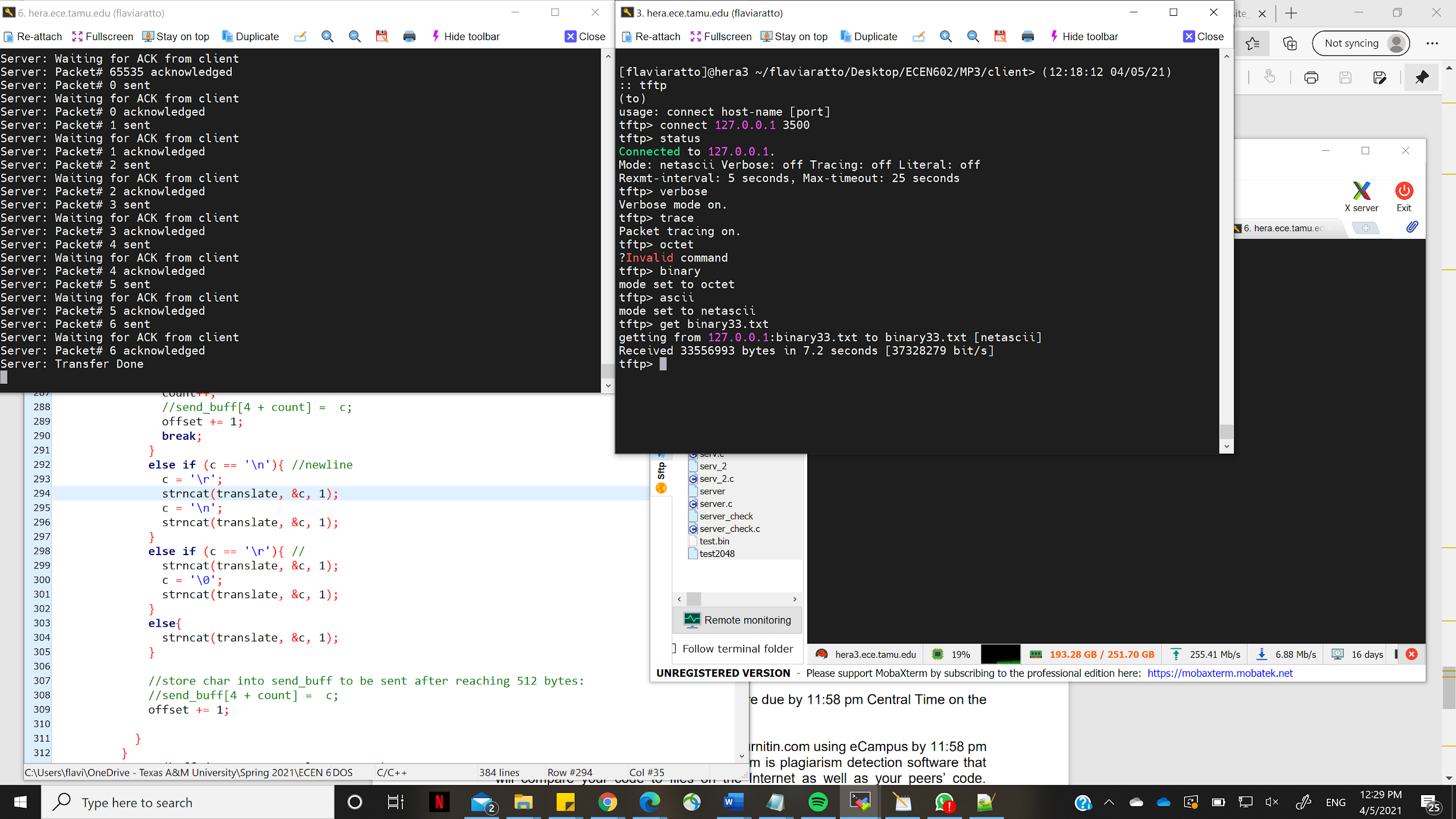
Result:

The resulting text file that the client received added ‘\0’ after each CR as noted in the assignment document:



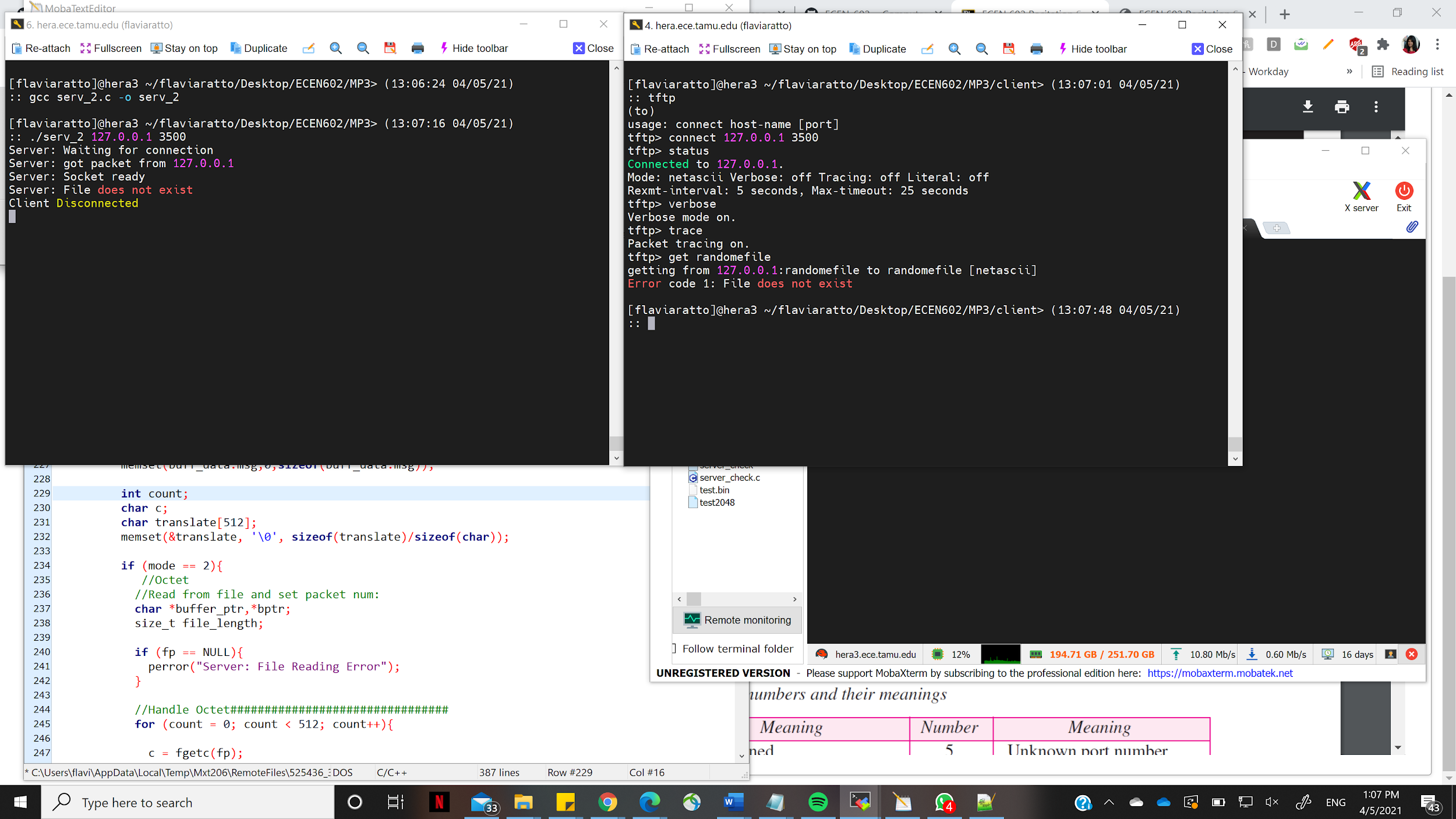
1. **Transfer a binary file of 34 MB and see if block number wrap-around works**

We generated a large binary file of 34 MB - “binary33.txt” as shown in the TA’s recitation slide. We see below that we were able to transfer the enter file and that the block number wrap-around after Packet # 65535.

****

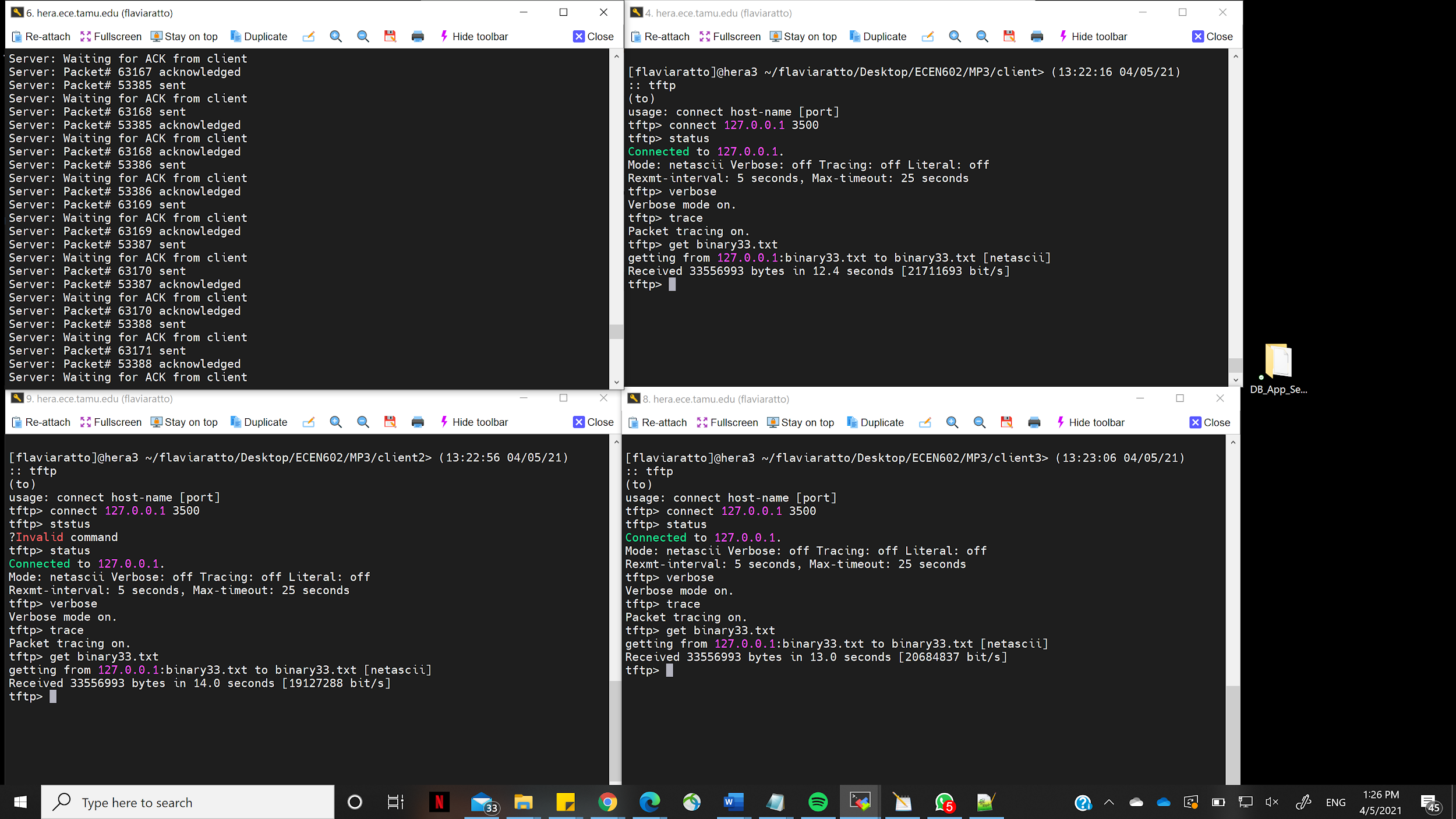
1. **Check that you receive an error message if you try to transfer a file that does not exist and that your server cleans up and the child process exits**

We tried to have the tftp client read a file from the server that does not exist. We see that error handling takes place and the server sends an error packet to the client.

****

1. **Connect to the TFTP server with three clients simultaneously and test that the transfers work correctly (you will probably need a big file to have them all running at the same time)**

We connected 3 clients and transferred a big file simultaneously and we see that the transfer works correctly.

****

1. **Terminate the TFTP client in the middle of a transfer and see if your TFTP server recognizes after 10 timeouts that the client is no longer there (you will need a big file)**

We made the client send an RRQ for a very big file. In the middle of the file transfer we made the client exit. We see that the server waits for acknowledgement and after timeout, it resends the packet to the client and waits for acknowledgement again. After 10 timeouts, the server reaches its maximum timeout and recognizes the client isn’t there.

