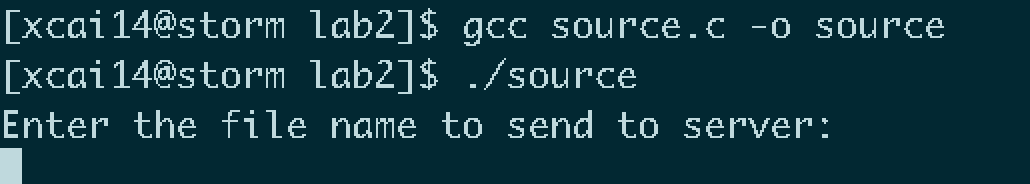
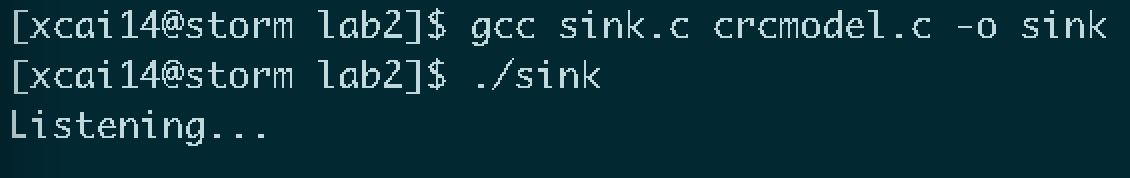
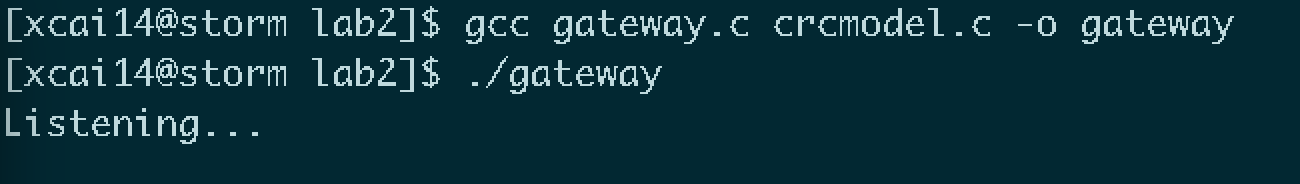
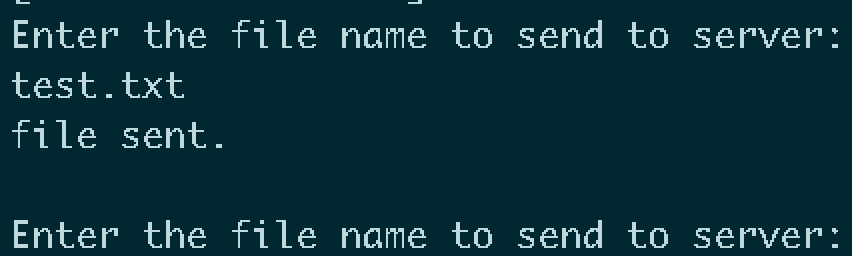
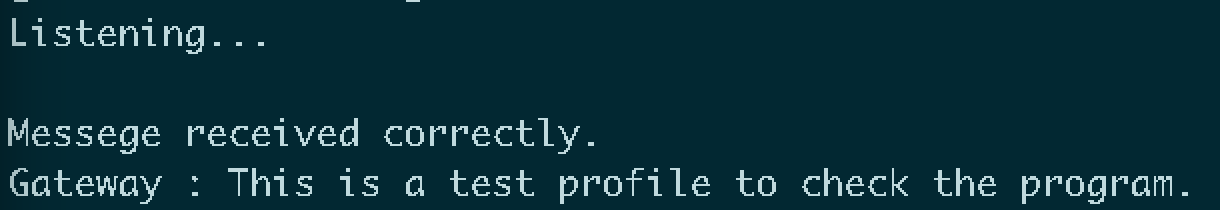
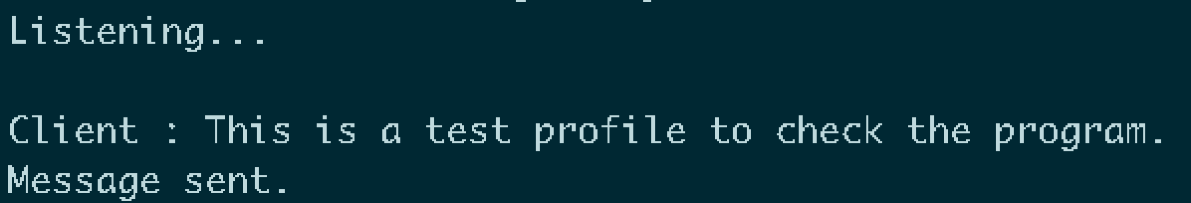
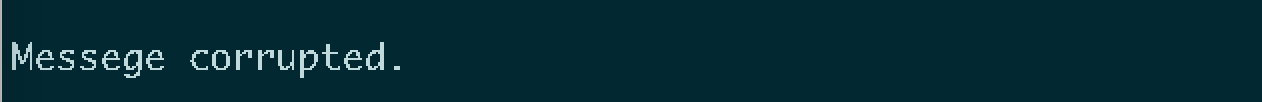
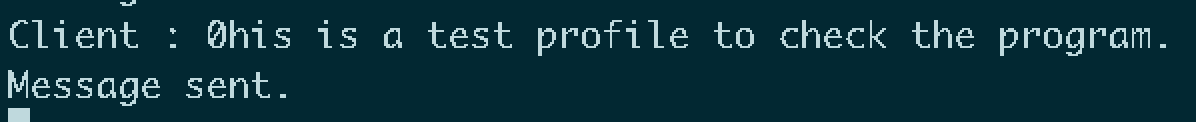
**Lab 2 Report**

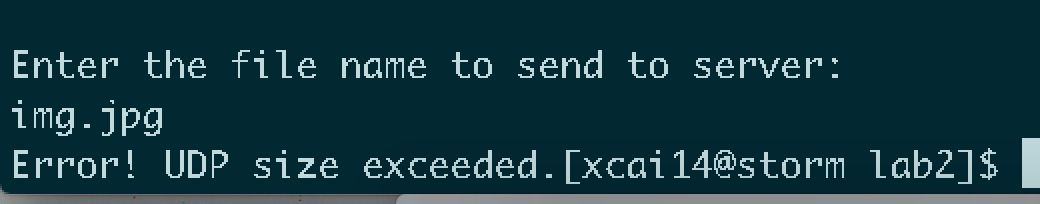
All three programs compile successfully along with Ross William’s crcmodel file. After compilation, the source asks the user to input the file name to transmit to gateway then to sink.

The first test is to transmit a standard txt file. The gateway is able to extract the string from the file, generate crc and concatenate it after the string. The sink is able to separate the crc from the string and then generate another crc from the formatted string then compare with the transmitted one. While they are equal, the sink will display the message sent from the client. 



The second test is to test when the data got corrupted during transition. The program builds a 20% chance in the gateway that it will change the first bit of the extracted string to 0. Consequently, after a few test, a message is altered by the program and then sink instantly detects it, reporting error. 

The third test was to check how the program handle other files which may possibly exceeds the size that UDP can handle. A large image file is going to be transmitted. While the source is extracting information, it detects that the size is larger than the UDP size which is 65530 bits(saved 5 bits to attach crc). The source reports error and exit.



Finally, the style of coding is pretty clear to follow from my perspective. It is a relatively short lab and I did comments on the parts that may be confusing to whoever may read the file especially the parts which uses the crc generic functions. I copied the user guide from the header file of the crc model to avoid unnecessary misunderstanding and Ross’s instruction is definitely clear enough for the user.