Students:

Ryan Ulgiati 7238561: responsible for implementing encryption/decryption, compression/decompression, integrity check, question 2

Nathan Hawrylak, 7268774 : Responsible for creating java server, using server sockets, threads, and the file system. Collected all of the data for Question 1 : Spent most time trying to implement Open Telemetry into the server however many attempts failed and could not get either maven or gradle to work.

Question 1:

Implementation :

To implement this server is very simple. Within the given file are 5 main components needed to run. There are 3 java classes and 2 folders: a load and a write. To begin if you are launching the client and the server from the same machine then the port that you will be using will be the local host port or 127.0.0.1 , otherwise you will need go to line 30 in Client java file using any text editor and replace the host “127.0.0.1” with the desired Ipv4 of the machine hosting the server. Once that is done open 2 cmd terminals one for the Server and one for the Client, in both terminals use cd command to move into the file directory storing the classes. Compile both classes using the javac command and then run using the java command. Make sure that the Server class in one terminal is run first before running the client class in the other. If you do not, the Client will throw an error. Once run the Client and Server will do what they need to do, taking all of the files from the load folder and storing them into the write folder. To check that it has worked, open the write folder and all of the files should be copied into it. If you would like to add more files to be copied, store them into the load file and run the client again. If you are running the server on a separate machine you will need to make sure that you change the path that the Server will write to otherwise it may throw an error. To solve this problem simply create a new folder wherever you like on your system and copy the file path. Then go to the Server class and paste the path into the double quotes on line 15 where the variable storage path is made. Depending on which IDE you are using, if you copy the path and paste it, it may not like the \ you are using, simply fix this by reversing it to / and it will be ok. **Note** please unless you are using the server on a separate machine, keep the folder together and do not move any pieces around. Also if you decide to move the Server file out make sure to bring the ClientHandler class with it into the same directory.

Performance analysis and Findings :

For our output data we can see many things. First, is that every time a file is transferred over to the server a new print line is outputted to the command line to say

that the storing job has been started. Next we can see that the server checks the integrity of the file by comparing its old size to its new one to make sure nothing has been added. The last part we can notice is the entry, leave, and total times. This is the main piece of data we will be analysing and working with. Start time is the time in milliseconds that the file started the transfer on the client side, leave time is the time after the file has been worked with and stored into the folder, and lastly the total time is calculated by taking leave time - enter time. Total time is the exact time it took in milliseconds for the client to start the sending process to the end of the storing process. As you can see by looking at the test runs there is a massive difference in terms of with and without the advanced features. This can be seen through the given data and also while running the code. Without the advanced features the server is quickly able to receive and process requests in small amounts of time. With the advanced features it takes much longer to process and the overall time from start to finish far exceeds that of the prior test. This is most definitely due to all of the different features run on the test that were implemented. The 3 features we decided to implement are encryption/decryption, compression/decompression, and finally an integrity check. These 3 features heavily slow down the run time of the processing.

Server test run : Without advanced features

System running
printing file : text1.txt
File data : text1.txt || Transfer start time : 1700447134181 Transfer end time : 1700447134193 Total time for completion : 12
printing file : text11.txt
File data : text11.txt || Transfer start time : 1700447134189 Transfer end time : 1700447134233 Total time for completion : 44
printing file : text12.txt
printing file : text13.txt
File data : text12.txt || Transfer start time : 1700447134237 Transfer end time : 1700447134241 Total time for completion : 4
File data : text13.txt || Transfer start time : 1700447134241 Transfer end time : 1700447134245 Total time for completion : 4
printing file : text14.txt
printing file : text17.txt
printing file : text16.txt
printing file : text19.txt
printing file : text18.txt
printing file : text15.txt
File data : text16.txt || Transfer start time : 1700447134249 Transfer end time : 1700447134277 Total time for completion : 28
File data : text17.txt || Transfer start time : 1700447134249 Transfer end time : 1700447134277 Total time for completion : 28

File data : text14.txt || Transfer start time : 1700447134241 Transfer end time : 1700447134277 Total time for completion : 36
File data : text19.txt || Transfer start time : 1700447134253 Transfer end time : 1700447134281 Total time for completion : 28
printing file : text20.txt
printing file : text2.txt
File data : text15.txt || Transfer start time : 1700447134245 Transfer end time : 1700447134281 Total time for completion : 36
File data : text18.txt || Transfer start time : 1700447134253 Transfer end time : 1700447134281 Total time for completion : 28
File data : text20.txt || Transfer start time : 1700447134261 Transfer end time : 1700447134305 Total time for completion : 44
printing file : text4.txt
printing file : text3.txt
File data : text2.txt || Transfer start time : 1700447134257 Transfer end time : 1700447134285 Total time for completion : 28
File data : text4.txt || Transfer start time : 1700447134289 Transfer end time : 1700447134309 Total time for completion : 20
printing file : text5.txt
File data : text3.txt || Transfer start time : 1700447134285 Transfer end time : 1700447134313 Total time for completion : 28
printing file : text6.txt
printing file : text8.txt
File data : text5.txt || Transfer start time : 1700447134293 Transfer end time : 1700447134321 Total time for completion : 28
printing file : text7.txt
File data : text8.txt || Transfer start time : 1700447134317 Transfer end time : 1700447134325 Total time for completion : 8
File data : text6.txt || Transfer start time : 1700447134313 Transfer end time : 1700447134325 Total time for completion : 12
printing file : text9.txt
File data : text7.txt || Transfer start time : 1700447134317 Transfer end time : 1700447134329 Total time for completion : 12
File data : text9.txt || Transfer start time : 1700447134317 Transfer end time : 1700447134333 Total time for completion : 16

Server test run : Using advanced features

System running
printing file : text1.txt
printing file : text11.txt

File size is correct for file : text1.txt
File data : text1.txt || Transfer start time : 1700446125775 Transfer end time : 1700446125951 Total time for completion : 176
File size is correct for file : text11.txt
File data : text11.txt || Transfer start time : 1700446125859 Transfer end time : 1700446164786 Total time for completion : 38927
printing file : text12.txt
printing file : text13.txt
printing file : text14.txt
printing file : text15.txt
File size is correct for file : text13.txt
File data : text13.txt || Transfer start time : 1700446167191 Transfer end time : 1700446167707 Total time for completion : 516
File size is correct for file : text14.txt
File data : text14.txt || Transfer start time : 1700446167255 Transfer end time : 1700446167862 Total time for completion : 607
printing file : text16.txt
printing file : text17.txt
File size is correct for file : text16.txt
File data : text16.txt || Transfer start time : 1700446167890 Transfer end time : 1700446167954 Total time for completion : 64
printing file : text18.txt
File size is correct for file : text17.txt
File data : text17.txt || Transfer start time : 1700446167942 Transfer end time : 1700446168014 Total time for completion : 72
printing file : text19.txt
File size is correct for file : text18.txt
File data : text18.txt || Transfer start time : 1700446167966 Transfer end time : 1700446168342 Total time for completion : 376
File size is correct for file : text15.txt
File data : text15.txt || Transfer start time : 1700446167387 Transfer end time : 1700446168358 Total time for completion : 971
File size is correct for file : text12.txt
File data : text12.txt || Transfer start time : 1700446164790 Transfer end time : 1700446168942 Total time for completion : 4152
printing file : text2.txt
printing file : text20.txt
File size is correct for file : text2.txt
File data : text2.txt || Transfer start time : 1700446169901 Transfer end time : 1700446169937 Total time for completion : 36
File size is correct for file : text19.txt
File data : text19.txt || Transfer start time : 1700446168234 Transfer end time : 1700446170885 Total time for completion : 2651
printing file : text3.txt

printing file : text4.txt
printing file : text5.txt
File size is correct for file : text4.txt
File data : text4.txt || Transfer start time : 1700446180233 Transfer end time : 1700446180293 Total time for completion : 60
File size is correct for file : text3.txt
File data : text3.txt || Transfer start time : 1700446179057 Transfer end time : 1700446182059 Total time for completion : 3002
printing file : text6.txt
printing file : text7.txt
File size is correct for file : text6.txt
File data : text6.txt || Transfer start time : 1700446185995 Transfer end time : 1700446186155 Total time for completion : 160
printing file : text8.txt
File size is correct for file : text7.txt
File data : text7.txt || Transfer start time : 1700446186071 Transfer end time : 1700446186695 Total time for completion : 624
printing file : text9.txt
File size is correct for file : text8.txt
File data : text8.txt || Transfer start time : 1700446186327 Transfer end time : 1700446187700 Total time for completion : 1373
File size is correct for file : text9.txt
File data : text9.txt || Transfer start time : 1700446186939 Transfer end time : 1700446188544 Total time for completion : 1605
File size is correct for file : text20.txt
File data : text20.txt || Transfer start time : 1700446169917 Transfer end time : 1700446193518 Total time for completion : 23601
File size is correct for file : text5.txt
File data : text5.txt || Transfer start time : 1700446180241 Transfer end time : 1700446197544 Total time for completion : 17303

Challenges Faced

The main challenge for this assignment was trying to get Open telemetry to work. Which as you can see, in the end I could not get to work. The main problem came in setting up the dependencies for the the examples to work, no matter how hard I tried maven would never properly set up for me, And when I tried to set up gradle I got closer however it ended up being worthless as it would prove to share same results with maven.

Question 2:

The bug inserted into the code was an extra step in the encryption method. If ‘X’ was detected as the current character, the char value would be decremented by 1 instead of being encrypted the proper way. This would cause the integrity checksum on the server side to fail because the text would be decrypted the standard way without accounting for the ‘X’ exception. When tested this way, the following results were recorded:

ClientData:

text1.txt: input[x] < 127-k && input[x] > 32
text1.txt: input[x] >= 127 || input[x] <= 32
text1.txt: input[x] < 127 && input[x] > 32

text2.txt: input[x] < 127-k && input[x] > 32
text2.txt: input[x] >= 127 || input[x] <= 32
text2.txt: input[x] == X

text3.txt: input[x] < 127-k && input[x] > 32
text3.txt: input[x] >= 127 || input[x] <= 32

text4.txt: input[x] == X
text4.txt: input[x] < 127-k && input[x] > 32
text4.txt: input[x] >= 127 || input[x] <= 32

text5.txt: input[x] < 127-k && input[x] > 32
text5.txt: input[x] >= 127 || input[x] <= 32
text5.txt: input[x] < 127 && input[x] > 32

text6.txt: input[x] < 127-k && input[x] > 32
text6.txt: input[x] >= 127 || input[x] <= 32
text6.txt: input[x] < 127 && input[x] > 32

text7.txt: input[x] < 127-k && input[x] > 32
text7.txt: input[x] >= 127 || input[x] <= 32

text8.txt: input[x] < 127-k && input[x] > 32
text8.txt: input[x] >= 127 || input[x] <= 32

text9.txt: input[x] < 127-k && input[x] > 32
text9.txt: input[x] >= 127 || input[x] <= 32

text10.txt: input[x] < 127-k && input[x] > 32
text10.txt: input[x] >= 127 || input[x] <= 32

text11.txt: input[x] < 127-k && input[x] > 32
text11.txt: input[x] >= 127 || input[x] <= 32

text12.txt: input[x] < 127-k && input[x] > 32
text12.txt: input[x] >= 127 || input[x] <= 32

text13.txt: input[x] < 127-k && input[x] > 32
text13.txt: input[x] >= 127 || input[x] <= 32
text13.txt: input[x] == X

text14.txt: input[x] == X
text14.txt: input[x] < 127-k && input[x] > 32
text14.txt: input[x] >= 127 || input[x] <= 32

text15.txt: input[x] < 127-k && input[x] > 32
text15.txt: input[x] >= 127 || input[x] <= 32

text16.txt: input[x] < 127-k && input[x] > 32
text16.txt: input[x] >= 127 || input[x] <= 32
text16.txt: input[x] == X

text17.txt: input[x] < 127-k && input[x] > 32
text17.txt: input[x] >= 127 || input[x] <= 32
text17.txt: input[x] == X
text17.txt: input[x] < 127 && input[x] > 32

text18.txt: input[x] < 127-k && input[x] > 32
text18.txt: input[x] >= 127 || input[x] <= 32
text18.txt: input[x] == X

text19.txt: input[x] < 127-k && input[x] > 32
text19.txt: input[x] >= 127 || input[x] <= 32

text20.txt: input[x] < 127 && input[x] > 32
text20.txt: input[x] < 127-k && input[x] > 32
text20.txt: input[x] >= 127 || input[x] <= 32

ServerData:

text1.txt: input[x] < 127 && input[x] > 32
text1.txt: (input[x] - k) >= 33
text1.txt: (input[x] - k) < 33

File size is correct for file : text1.txt

text2.txt: input[x] < 127 && input[x] > 32
text2.txt: (input[x] - k) >= 33
File size is incorrect for file : text2.txt

text3.txt: input[x] < 127 && input[x] > 32
text3.txt: (input[x] - k) >= 33
File size is correct for file : text3.txt

text4.txt: input[x] < 127 && input[x] > 32
text4.txt: (input[x] - k) >= 33
File size is incorrect for file : text4.txt

text5.txt: input[x] < 127 && input[x] > 32
text5.txt: (input[x] - k) >= 33
File size is correct for file : text5.txt

text6.txt: input[x] < 127 && input[x] > 32
text6.txt: (input[x] - k) >= 33
text6.txt: (input[x] - k) < 33
File size is correct for file : text6.txt

text7.txt: input[x] < 127 && input[x] > 32
text7.txt: (input[x] - k) >= 33
File size is correct for file : text7.txt

text8.txt: input[x] < 127 && input[x] > 32
text8.txt: (input[x] - k) >= 33
File size is correct for file : text8.txt

text9.txt: input[x] < 127 && input[x] > 32
text9.txt: (input[x] - k) >= 33
File size is correct for file : text9.txt

text10.txt: input[x] < 127 && input[x] > 32
text10.txt: (input[x] - k) >= 33
File size is correct for file : text10.txt

text11.txt: input[x] < 127 && input[x] > 32
text11.txt: (input[x] - k) >= 33
File size is correct for file : text11.txt

text12.txt: input[x] < 127 && input[x] > 32

text12.txt: (input[x] - k) >= 33
File size is correct for file : text12.txt

text13.txt: input[x] < 127 && input[x] > 32
text13.txt: (input[x] - k) >= 33
File size is incorrect for file : text13.txt

text14.txt: input[x] < 127 && input[x] > 32
text14.txt: (input[x] - k) >= 33
File size is incorrect for file : text14.txt

text15.txt: input[x] < 127 && input[x] > 32
text15.txt: (input[x] - k) >= 33
File size is correct for file : text15.txt

text16.txt: input[x] < 127 && input[x] > 32
text16.txt: (input[x] - k) >= 33
File size is incorrect for file : text16.txt

text17.txt: input[x] < 127 && input[x] > 32
text17.txt: (input[x] - k) >= 33
text17.txt: (input[x] - k) < 33
File size is incorrect for file : text17.txt

text18.txt: input[x] < 127 && input[x] > 32
text18.txt: (input[x] - k) >= 33
File size is incorrect for file : text18.txt

text19.txt: input[x] < 127 && input[x] > 32
text19.txt: (input[x] - k) >= 33
File size is correct for file : text19.txt

text20.txt: input[x] < 127 && input[x] > 32
text20.txt: (input[x] - k) < 33
text20.txt: (input[x] - k) >= 33
File size is correct for file : text20.txt

Using this data, I was able to calculate the proper metrics for statistical debugging such as failure, context, and increase of each of the predicates to analyse the potential causes of the bug.

	text1	text2	text3	text4	text5	text6	text7	text8	text9	text10	text11	text12	text13	text14	text15	text16	text17	text18	text19	text20
Client: input[x] < 127-k && input[x] > 32	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
Client: input[x] >= 127 || input[x] <= 32	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
Client: input[x] == X	0	*	0	*	0	0	0	0	0	0	0	0	*	*	0	*	*	*	0	0
Client: input[x] < 127 && input[x] > 32	*	0	0	0	*	*	0	0	0	0	0	0	0	0	0	0	*	0	0	*
Server: input[x] < 127 && input[x] > 32	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
Server: (input[x] - k) >= 33	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
Server: (input[x] - k) < 33	*	0	0	0	0	*	0	0	0	0	0	0	0	0	0	0	*	0	0	*
Server: S/F	S	F	S	F	S	S	S	S	S	S	S	S	F	F	S	F	F	F	S	S

	F(P)	S(P)	Failure(P)	Context(P)	Increase(P)
P1: Client: input[x] < 127-k && input[x] > 32	7	13	0.35	0.35	0
P2: Client: input[x] >= 127 || input[x] <= 32	7	13	0.35	0.35	0
P3: Client: input[x] == X	7	0	1	0.35	0.65
P4: Client: input[x] < 127 && input[x] > 32	1	4	0.2	0.35	-0.15
P5: Server: input[x] < 127 && input[x] > 32	7	13	0.35	0.35	0
P6: Server: (input[x] - k) >= 33	7	13	0.35	0.35	0
P7: Server: (input[x] - k) < 33	1	4	0.2	0.35	-0.15

Due to negative increases being ignored and zeros being constants that were always true, there is only one possible option that the bug could have been which was indeed the line that checked if the character was ‘X’ with a failure of 1.00, a context of 0.35, and an increase of 0.65. This was the error that was inserted and has now been properly identified using statistical debugging so that it can be removed from the program.

Question 2 Bonus:

Now that the bug was identified, it was removed from the program. The line where the character checked for ‘X’ was left in for comparisons but it was now treated as it should be in the encryption program so that when decrypted, it would no longer cause this bug. Upon fixing the bug so that ‘X’ was treated normally, the following results were recorded:

	text1	text2	text3	text4	text5	text6	text7	text8	text9	text10	text11	text12	text13	text14	text15	text16	text17	text18	text19	text20
Client: input[x] < 127-k && input[x] > 32	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
Client: input[x] >= 127 || input[x] <= 32	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
Client: input[x] == X	0	*	0	*	0	0	0	0	0	0	0	0	*	*	0	*	*	*	0	0
Client: input[x] < 127 && input[x] > 32	*	0	0	0	*	*	0	0	0	0	0	0	0	0	0	0	*	0	0	*
Server: input[x] < 127 && input[x] > 32	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
Server: (input[x] - k) >= 33	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
Server: (input[x] - k) < 33	*	0	0	0	0	*	0	0	0	0	0	0	0	0	0	0	*	0	0	*
Server: S/F	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S

	F(P)	S(P)	Failure(P)	Context(P)	Increase(P)
P1: Client: input[x] < 127-k && input[x] > 32	0	20	0	0	0
P2: Client: input[x] >= 127 || input[x] <= 32	0	20	0	0	0
P3: Client: input[x] == X	0	7	0	0	0
P4: Client: input[x] < 127 && input[x] > 32	0	5	0	0	0
P5: Server: input[x] < 127 && input[x] > 32	0	20	0	0	0
P6: Server: (input[x] - k) >= 33	0	20	0	0	0
P7: Server: (input[x] - k) < 33	0	5	0	0	0

Now that the bug has been removed, all predicates have a failure, context, and increase of 0 because there is no data that contains any bugs for any of the predicates.