Task 1

Using eth0 because eth1 keeps giving error message ‘ssh: connect to host 10.0.2.15 port 22: Connection timed out’

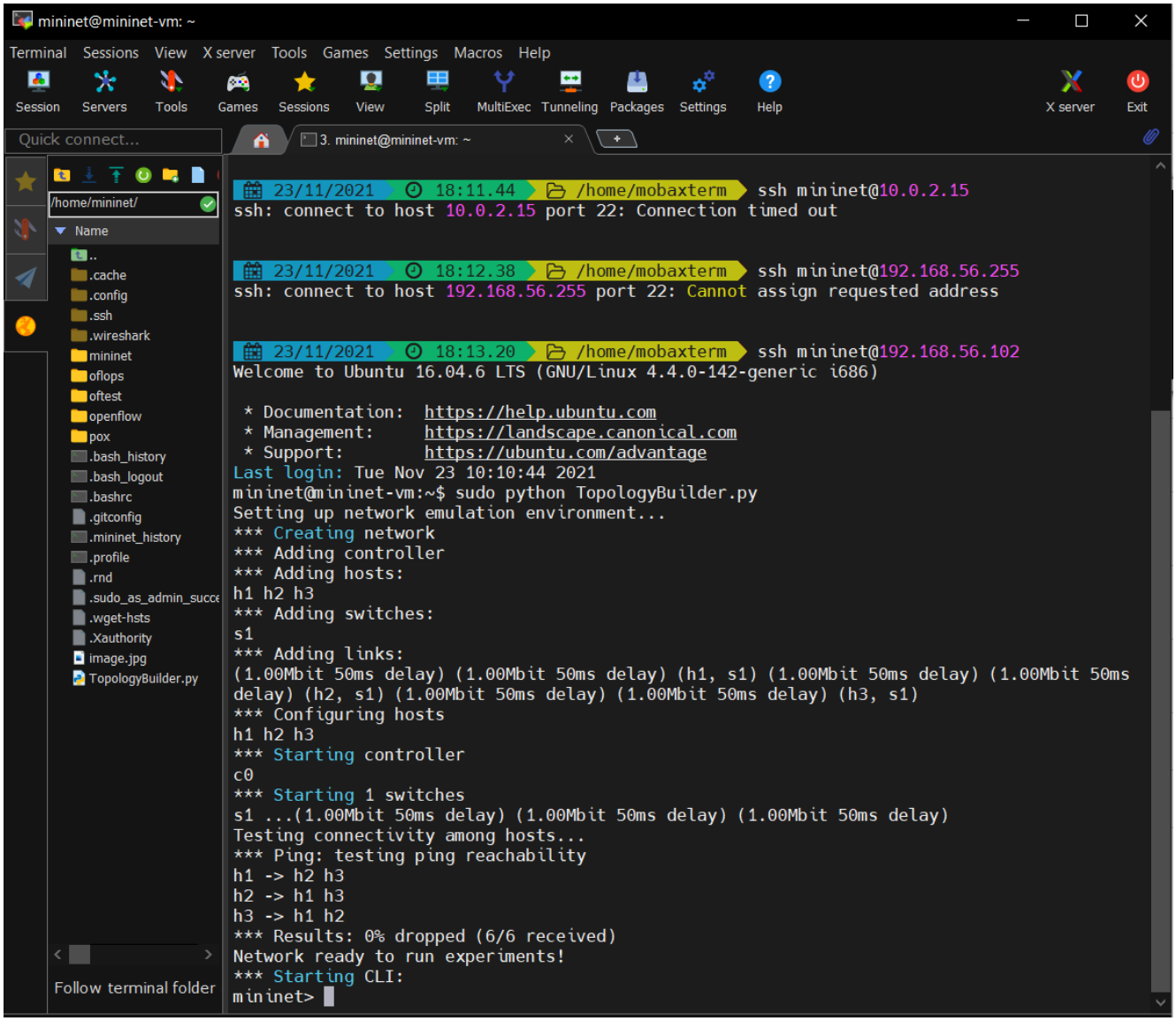


Figure 1: Task1

Task 2

1-2.

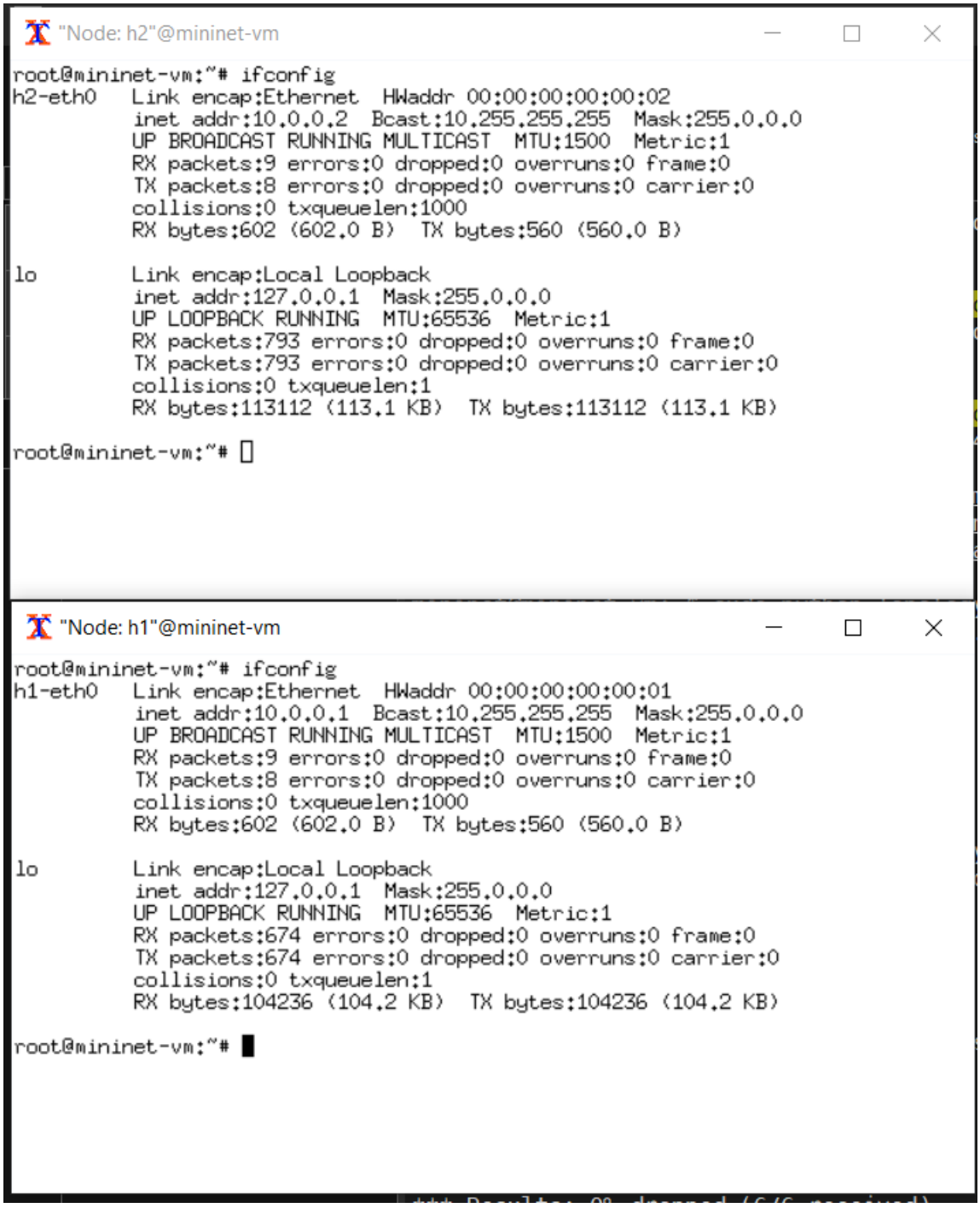


Figure 2:Task 2 steps 1 and 2

Q1. It took 4.312606059 ms to transfer the file. All UDP packages are moving in one direction from h1 to h2 as shown in the image below. This is because the transferring of a file does not require any data from the client since UDP does not require a handshake to connect.

244	3.968632012	10.0.0.1	10.0.0.2	UDP	2090 54132 → 34567 Len=2048
245	3.985113905	10.0.0.1	10.0.0.2	UDP	2090 54132 → 34567 Len=2048
246	4.001295341	10.0.0.1	10.0.0.2	UDP	2090 54132 → 34567 Len=2048
247	4.017672305	10.0.0.1	10.0.0.2	UDP	2090 54132 → 34567 Len=2048
248	4.034121376	10.0.0.1	10.0.0.2	UDP	2090 54132 → 34567 Len=2048
249	4.050612136	10.0.0.1	10.0.0.2	UDP	2090 54132 → 34567 Len=2048
250	4.067176795	10.0.0.1	10.0.0.2	UDP	2090 54132 → 34567 Len=2048
251	4.083216493	10.0.0.1	10.0.0.2	UDP	2090 54132 → 34567 Len=2048
252	4.099617683	10.0.0.1	10.0.0.2	UDP	2090 54132 → 34567 Len=2048
253	4.116178875	10.0.0.1	10.0.0.2	UDP	2090 54132 → 34567 Len=2048
254	4.132362965	10.0.0.1	10.0.0.2	UDP	2090 54132 → 34567 Len=2048
255	4.148833467	10.0.0.1	10.0.0.2	UDP	2090 54132 → 34567 Len=2048
256	4.165170377	10.0.0.1	10.0.0.2	UDP	2090 54132 → 34567 Len=2048
257	4.181816505	10.0.0.1	10.0.0.2	UDP	2090 54132 → 34567 Len=2048
258	4.198568152	10.0.0.1	10.0.0.2	UDP	2090 54132 → 34567 Len=2048
259	4.214347668	10.0.0.1	10.0.0.2	UDP	2090 54132 → 34567 Len=2048
260	4.230820814	10.0.0.1	10.0.0.2	UDP	2090 54132 → 34567 Len=2048
261	4.247474582	10.0.0.1	10.0.0.2	UDP	2090 54132 → 34567 Len=2048
262	4.263505443	10.0.0.1	10.0.0.2	UDP	2090 54132 → 34567 Len=2048
263	4.279975053	10.0.0.1	10.0.0.2	UDP	2090 54132 → 34567 Len=2048
264	4.296723909	10.0.0.1	10.0.0.2	UDP	2090 54132 → 34567 Len=2048
265	4.312606059	10.0.0.1	10.0.0.2	UDP	657 54132 → 34567 Len=615

Figure 3:Task2_Q1

Q2

The received image is the same and it is of the same size as the original test file.

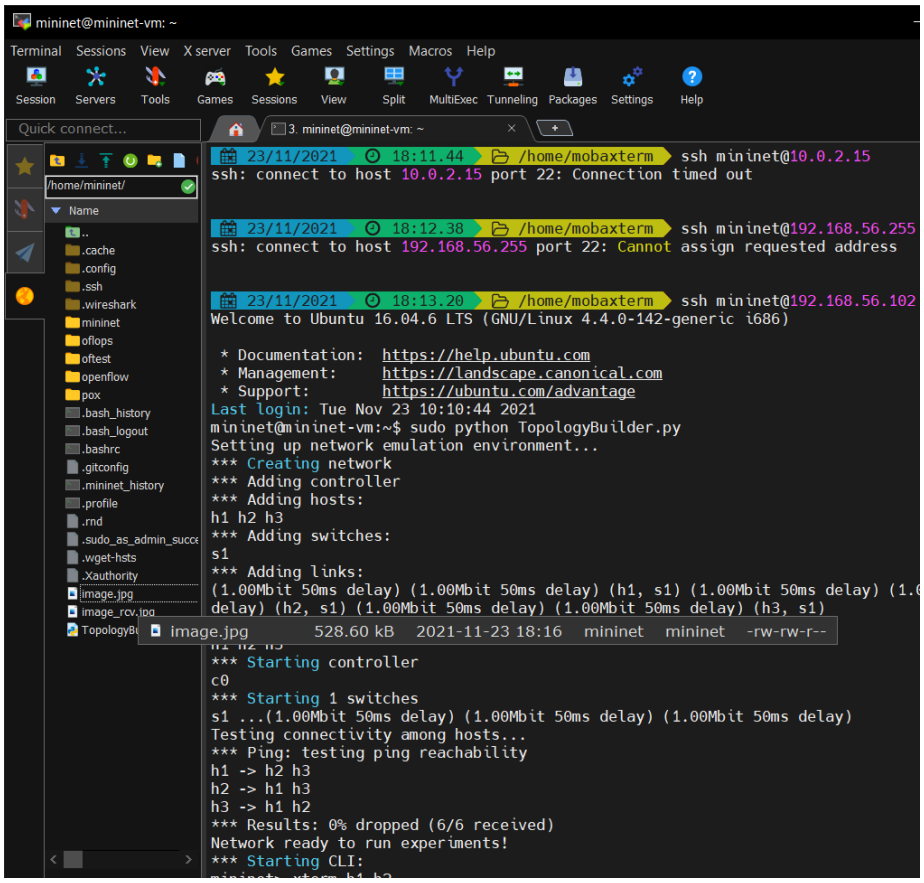


Figure 4: Recieve image size befor transfer. Q2 Task2

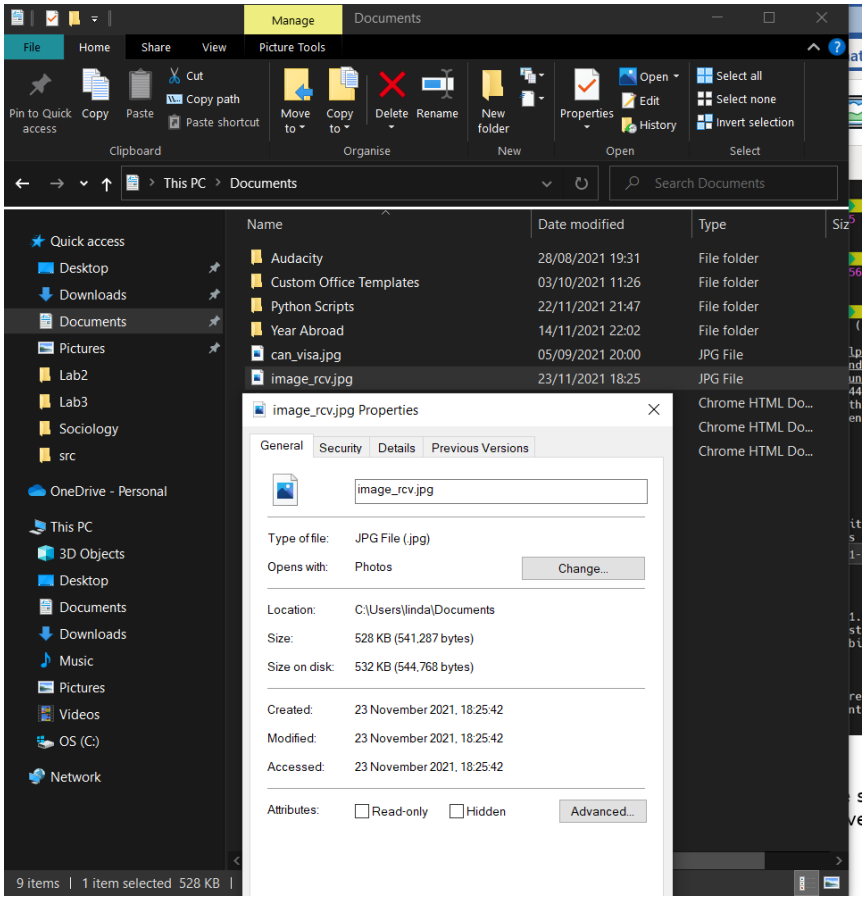


Figure 5:Task2_Q2 Image after transfer to local OS

Q3.

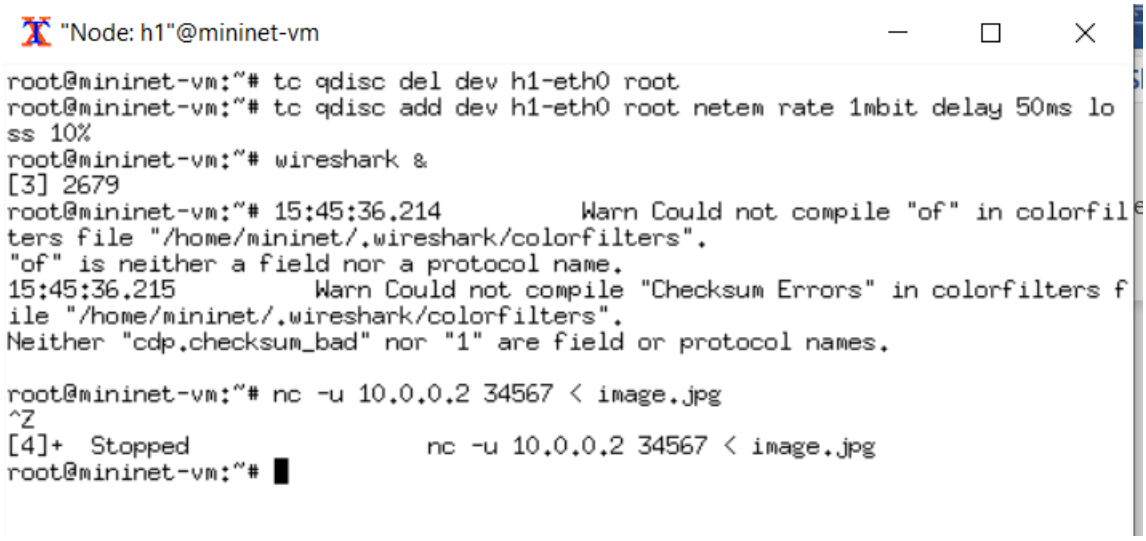


Figure 6:Task2_Q3

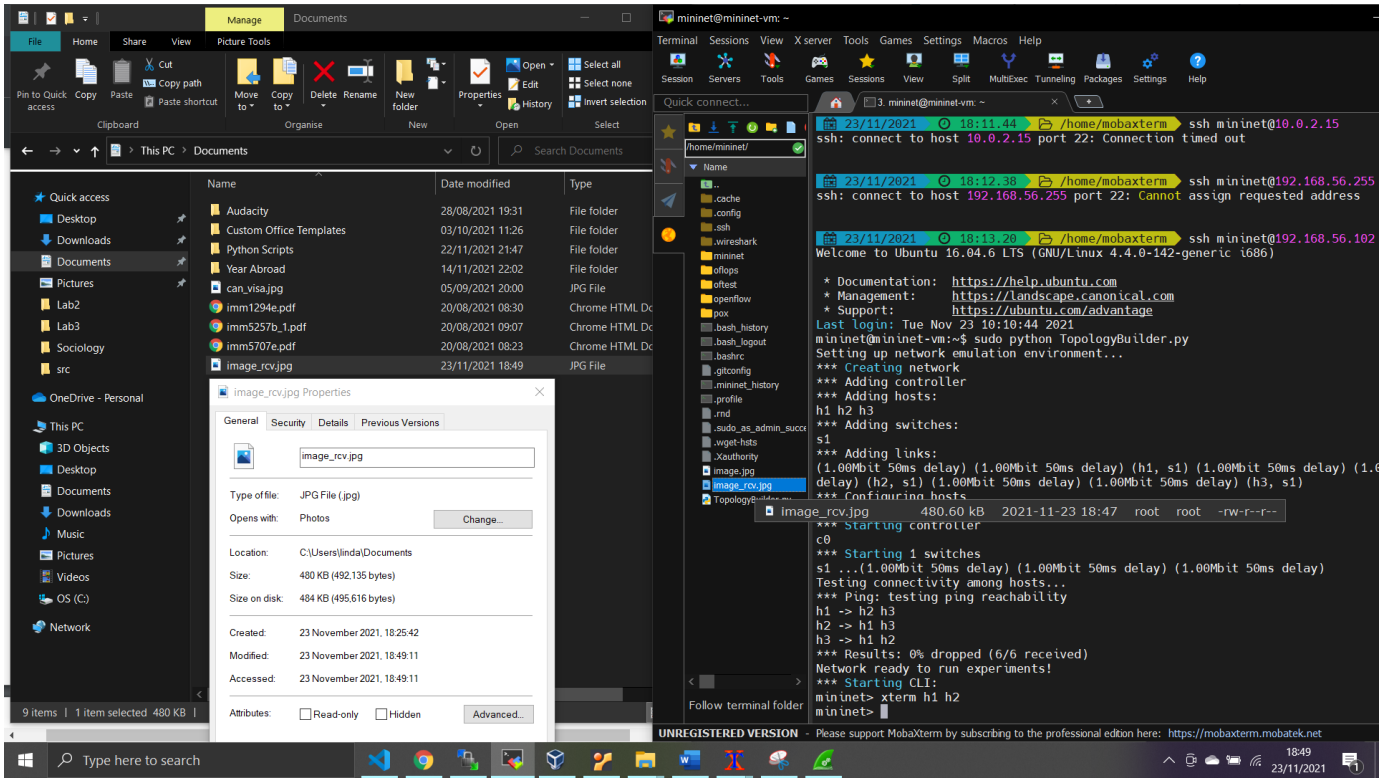


Figure 7: Task2_Q3

Yes, firstly the file took less time to be transferred as seen in Figure 8Figure 8:Task2_Q3 and the received file is smaller and corrupted as it can not be opened in my host OS.

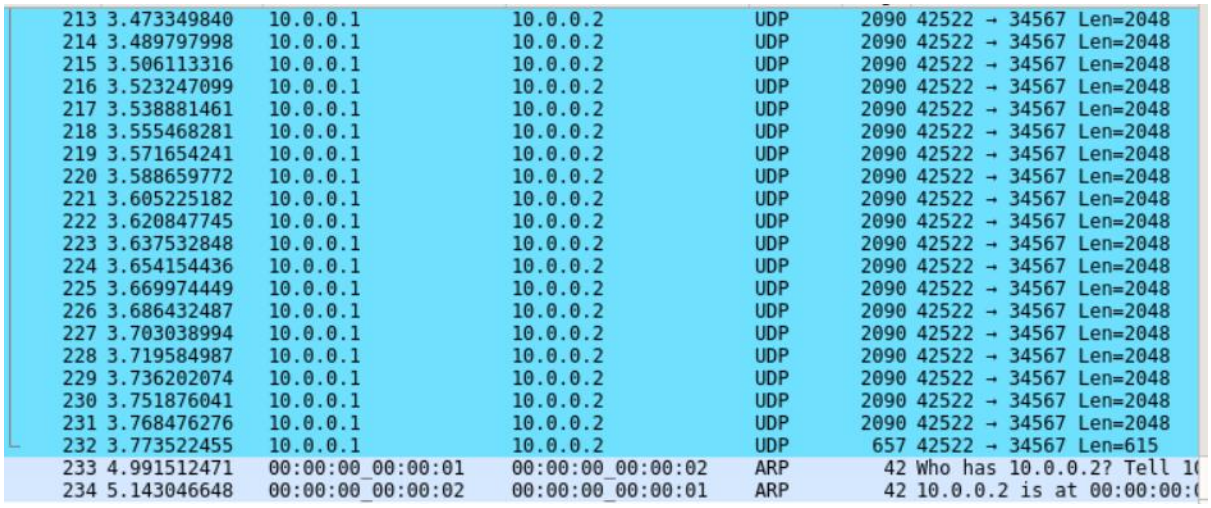


Figure 8:Task2_Q3

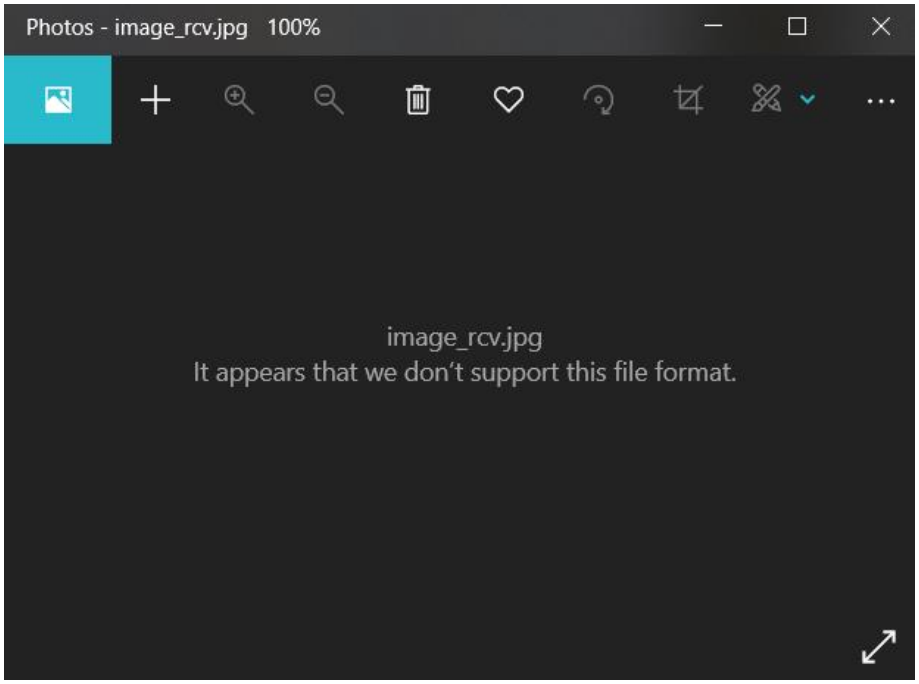


Figure 9:Task2_Q3 received file can't be opened

Task 3

Q4. It took approximately 5ms and there are packets being transferred from both directions. This is because a TCP connection requires a handshake and checks to ensure that all data has been received then closes the connection. This requires information to be sent back and forth from the server and the client as it is a two way communication.

No.	Time	Source	Destination	Protocol	Length	Info
364	4.690835072	10.0.0.1	10.0.0.2	TCP	2962	48260 → 34567 [ACK]
365	4.697661681	10.0.0.2	10.0.0.1	TCP	66	34567 → 48260 [ACK]
366	4.715214160	10.0.0.1	10.0.0.2	TCP	2962	48260 → 34567 [ACK]
367	4.720531630	10.0.0.2	10.0.0.1	TCP	66	34567 → 48260 [ACK]
368	4.727074152	10.0.0.1	10.0.0.2	TCP	1514	48260 → 34567 [ACK]
369	4.746519355	10.0.0.2	10.0.0.1	TCP	66	34567 → 48260 [ACK]
370	4.751422131	10.0.0.1	10.0.0.2	TCP	2962	48260 → 34567 [ACK]
371	4.770946715	10.0.0.2	10.0.0.1	TCP	66	34567 → 48260 [ACK]
372	4.775843159	10.0.0.1	10.0.0.2	TCP	2962	48260 → 34567 [ACK]
373	4.788203566	10.0.0.1	10.0.0.2	TCP	1514	48260 → 34567 [ACK]
374	4.795510252	10.0.0.2	10.0.0.1	TCP	66	34567 → 48260 [ACK]
375	4.812097643	10.0.0.1	10.0.0.2	TCP	2962	48260 → 34567 [ACK]
376	4.817299444	10.0.0.2	10.0.0.1	TCP	66	34567 → 48260 [ACK]
377	4.836570375	10.0.0.1	10.0.0.2	TCP	2961	48260 → 34567 [FIN,
378	4.841830746	10.0.0.2	10.0.0.1	TCP	66	34567 → 48260 [ACK]
379	4.865895943	10.0.0.2	10.0.0.1	TCP	66	34567 → 48260 [ACK]
380	4.903069096	10.0.0.2	10.0.0.1	TCP	66	34567 → 48260 [ACK]
381	4.927699768	10.0.0.2	10.0.0.1	TCP	66	34567 → 48260 [ACK]
382	4.963109264	10.0.0.2	10.0.0.1	TCP	66	34567 → 48260 [ACK]
383	4.987631511	10.0.0.2	10.0.0.1	TCP	66	34567 → 48260 [ACK]
384	4.987633675	10.0.0.2	10.0.0.1	TCP	66	34567 → 48260 [FIN,
385	5.038464896	10.0.0.1	10.0.0.2	TCP	66	48260 → 34567 [ACK]

Figure 10:Task3_Q4

Q5. Yes, the received image is the same size as the original image. They are of the same size.

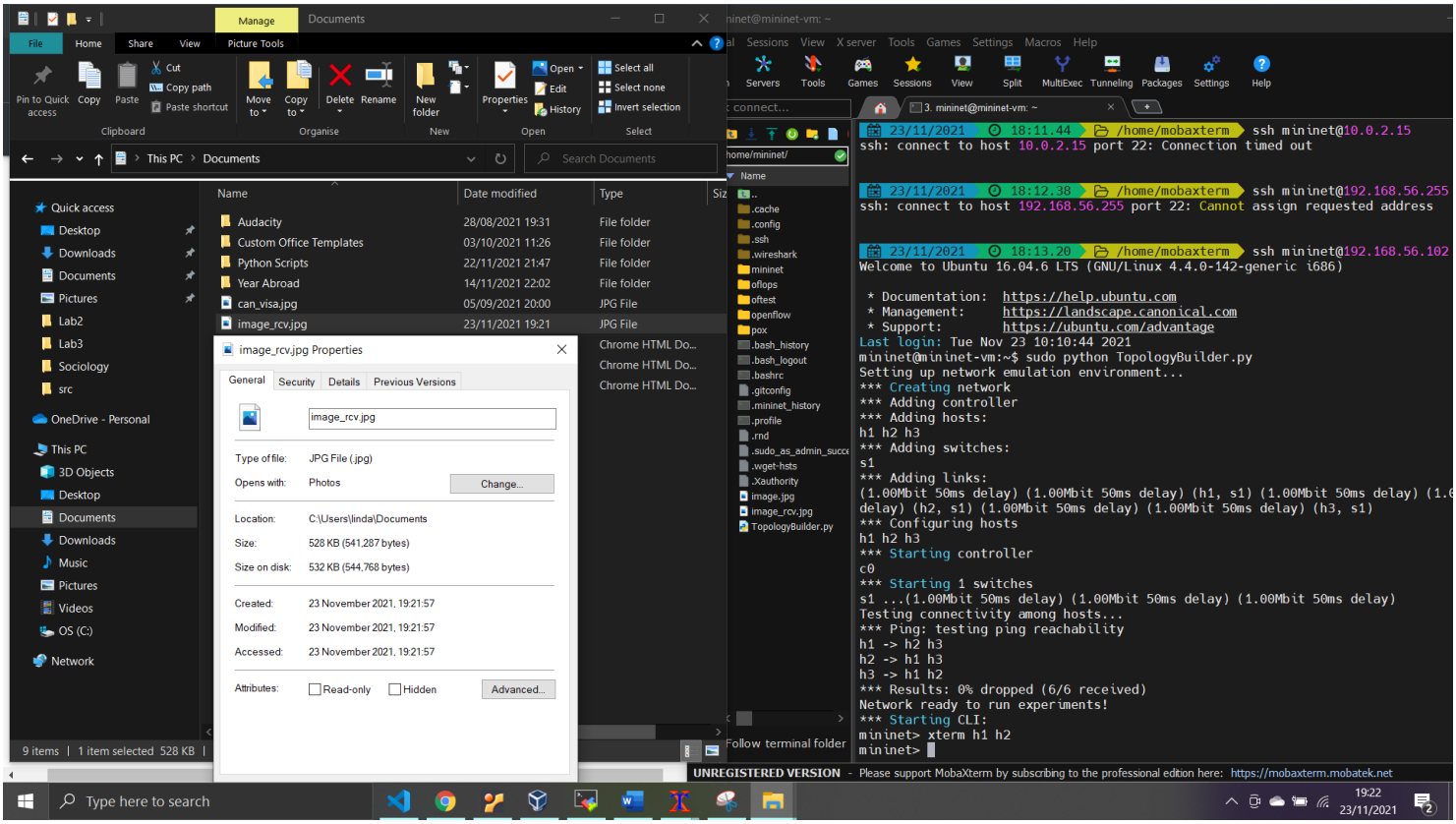


Figure 11: Task3_Q5

Q6. Yes, the transfer took approximately 15ms, which is about three times longer than before. However, the received image is of the same size as the original.

No.	Time	Source	Destination	Protocol	Length	Info
456	14.739772910	10.0.0.2	10.0.0.1	TCP	66	34567 → 48312 [ACK]
457	14.794770721	10.0.0.2	10.0.0.1	TCP	66	34567 → 48312 [ACK]
458	14.814422261	10.0.0.1	10.0.0.2	TCP	2962	[TCP Previous segmen
459	14.857573662	10.0.0.1	10.0.0.2	TCP	1514	[TCP Previous segmen
460	14.964590947	10.0.0.2	10.0.0.1	TCP	78	[TCP Dup ACK 457#1]
461	15.008842016	10.0.0.2	10.0.0.1	TCP	86	[TCP Dup ACK 457#2]
462	15.071452647	10.0.0.1	10.0.0.2	TCP	1514	[TCP Retransmission]
463	15.222499364	10.0.0.2	10.0.0.1	TCP	78	34567 → 48312 [ACK]
464	15.284982727	10.0.0.1	10.0.0.2	TCP	1514	[TCP Retransmission]
465	15.296848865	10.0.0.1	10.0.0.2	TCP	1514	48312 → 34567 [ACK]
466	15.435282420	10.0.0.2	10.0.0.1	TCP	66	34567 → 48312 [ACK]
467	15.497440402	10.0.0.1	10.0.0.2	TCP	1514	48312 → 34567 [ACK]
468	15.510178077	10.0.0.1	10.0.0.2	TCP	1514	48312 → 34567 [ACK]
469	15.510192515	10.0.0.2	10.0.0.1	TCP	66	34567 → 48312 [ACK]
470	15.573094823	10.0.0.1	10.0.0.2	TCP	1514	48312 → 34567 [ACK]
471	15.660939187	10.0.0.2	10.0.0.1	TCP	66	34567 → 48312 [ACK]
472	15.723797933	10.0.0.1	10.0.0.2	TCP	1514	48312 → 34567 [ACK]
473	15.735280158	10.0.0.1	10.0.0.2	TCP	1513	48312 → 34567 [FIN,
474	15.827587792	10.0.0.2	10.0.0.1	TCP	66	34567 → 48312 [ACK]
475	15.886328089	10.0.0.2	10.0.0.1	TCP	66	34567 → 48312 [ACK]
476	15.886330323	10.0.0.2	10.0.0.1	TCP	66	34567 → 48312 [FIN,
477	15.937290759	10.0.0.1	10.0.0.2	TCP	66	48312 → 34567 [ACK]

Figure 12:Task3_Q6

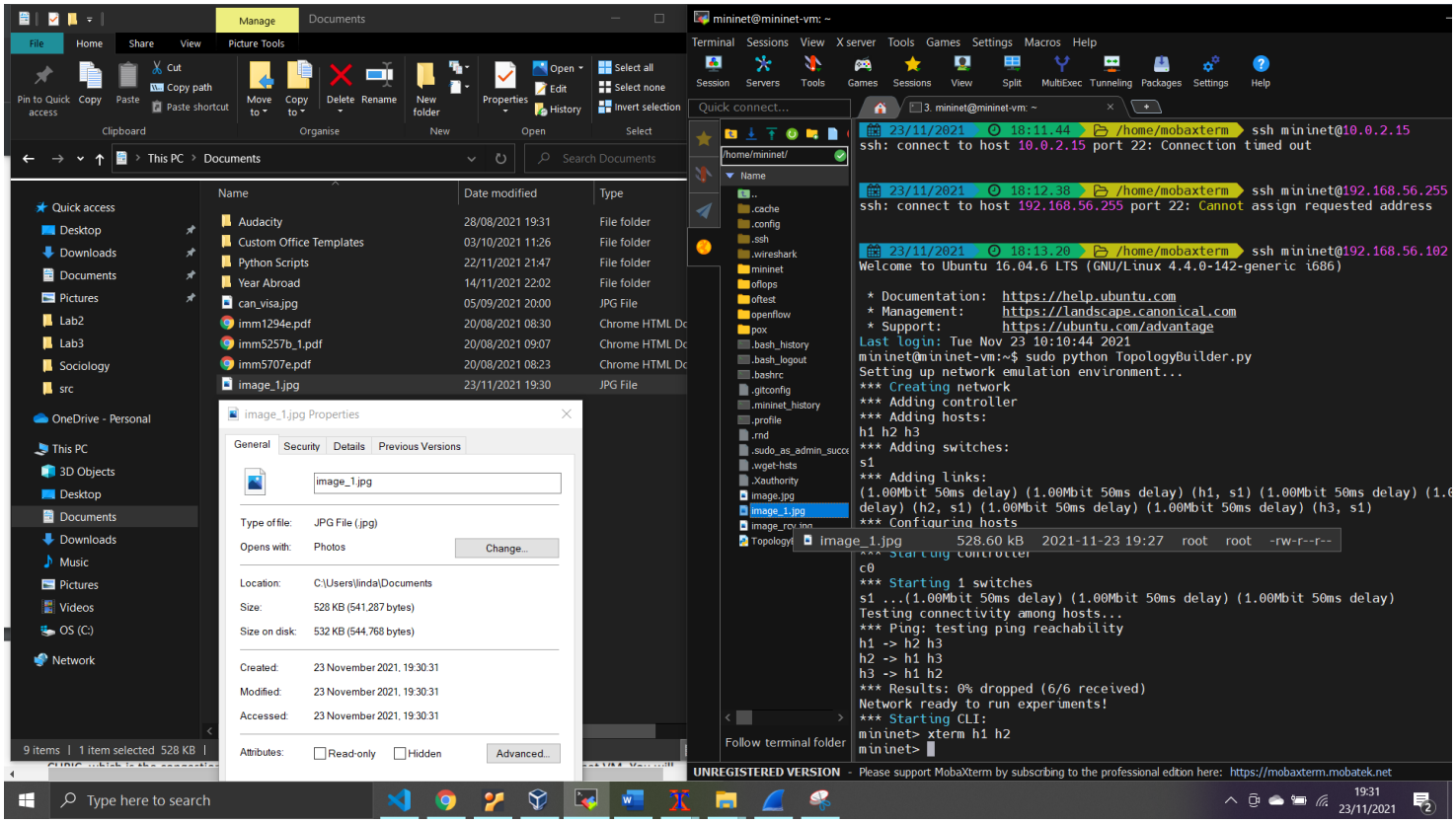


Figure 13:Task 3_Q6

Task 4

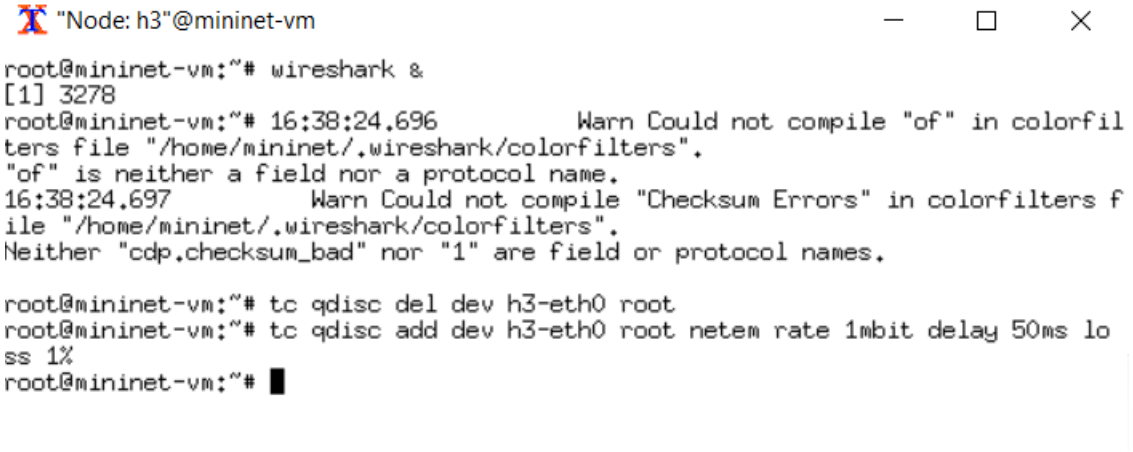


Figure 14:Task4

Q7.

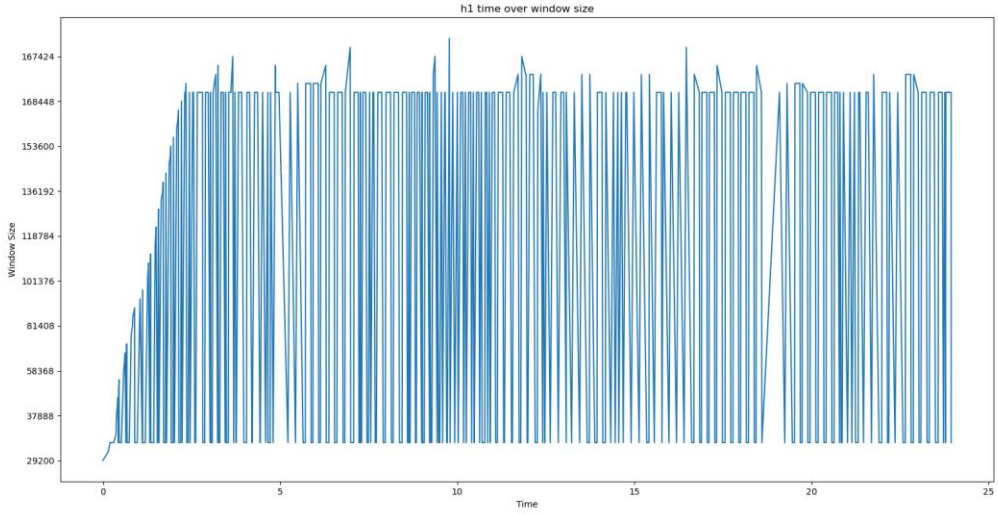


Figure 15: Task4_Q7. H1

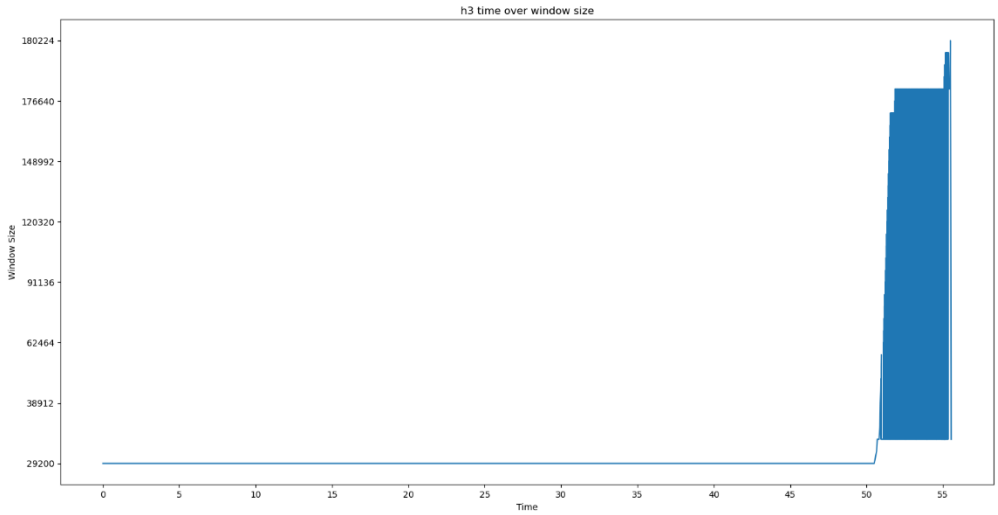


Figure 16:Task4_Q7. H3

Q8. The window size of the h1 file that has a 10% packet loss begins to fluctuate right from the beginning. It increases until it reaches a threshold then drops. While the window size of the h3 file that only has 1% packet loss remains constant throughout up until the end. This is because the TCP congestion control checks for lost acknowledgements and increases window size to enable operations that require more resources to be faster, when it reaches a threshold, it decreases. We also see that even though h3 only had a loss of 1% it took longer because the resources where being used up by h1.