Reliable Transport Protocol

CIE - 447

Beshoy Malak	201800613
Youssef Youssry	201901047
Ahmed Abdelgawad	201901249
Touransland Ali	202000398

Introduction & Objectives:

- The implementation of transport protocols that ensure reliable data transfer is crucial in modern computer networks. In this project, we will focus on augmenting the User Datagram Protocol (UDP) with the Go-Back-N (GBN) protocol to provide reliability services.
- The main objective of this project is to implement a special GBN sender and receiver that will provide reliable data transfer services over the unreliable UDP. Specifically, we aim to design and develop a protocol that will ensure that all data sent by the sender is received by the receiver without any loss, duplication or out of order delivery.

Brief:

As a brief, the protocol undergoes the following steps:

- To transmit a file to a receiver, the sender first divides the file into parts, each consisting of a specific number of bytes called the maximum segment size (MSS).
- The sender then sends packets to the receiver, which contain information such as packet and file IDs, application data, and a trailer.
- The sender sends a number of segments equal to the window size (N) and waits for acknowledgement from the receiver within a certain timeout.
- If the sender receives the acknowledgement before the timeout expires, it moves the window forward and sends new packets.
- If the sender does not receive the expected acknowledgement, it resends the packets in the window starting from the last one for which acknowledgement was not received.

Discussion:

The window size

In the GBN protocol, the window size is the parameter responsible for how many packets can be sent by the sender before waiting for the receiver's acknowledgement. Varying its size has many impacts on the system, as when we increase the window size, it improves the throughput of the protocol because it permits additional packets to be transmitted before waiting for the acknowledgment. But this advantage has a bad side effect, because larger window size also increases the risk of packet loss, this is because if an error occurs then many packets will be lost. Hence, choosing a window size doesn't only depend on how big the size is, but it also depends on other factors such as the error rate of the network and the bandwidth.

The window size affects the transfer rate so in our project when we intentionally used a higher window size, of value equals 15, so the average transfer rate increases. Hence, we can optimize our solution more efficiently.

The time-out interval

The time out determines the duration that the sender would wait for an acknowledgement. After this period of time, if the sender doesn't receive an acknowledgement, then it can assume that the packet is lost, so retransmitted again.

If we decrease the time out interval, then the system performance will certainly increase due to reducing the period that the sender waits until it detects a packet loss then retransmitted. But this increase in performance could have a bad side effect which is increasing the risk of false retransmission because not all delayed packets are always lost. Hence, in order to select a perfect time out interval, we should have an expectation regarding the delay in receiving acknowledgement and it also depends on the network latency.

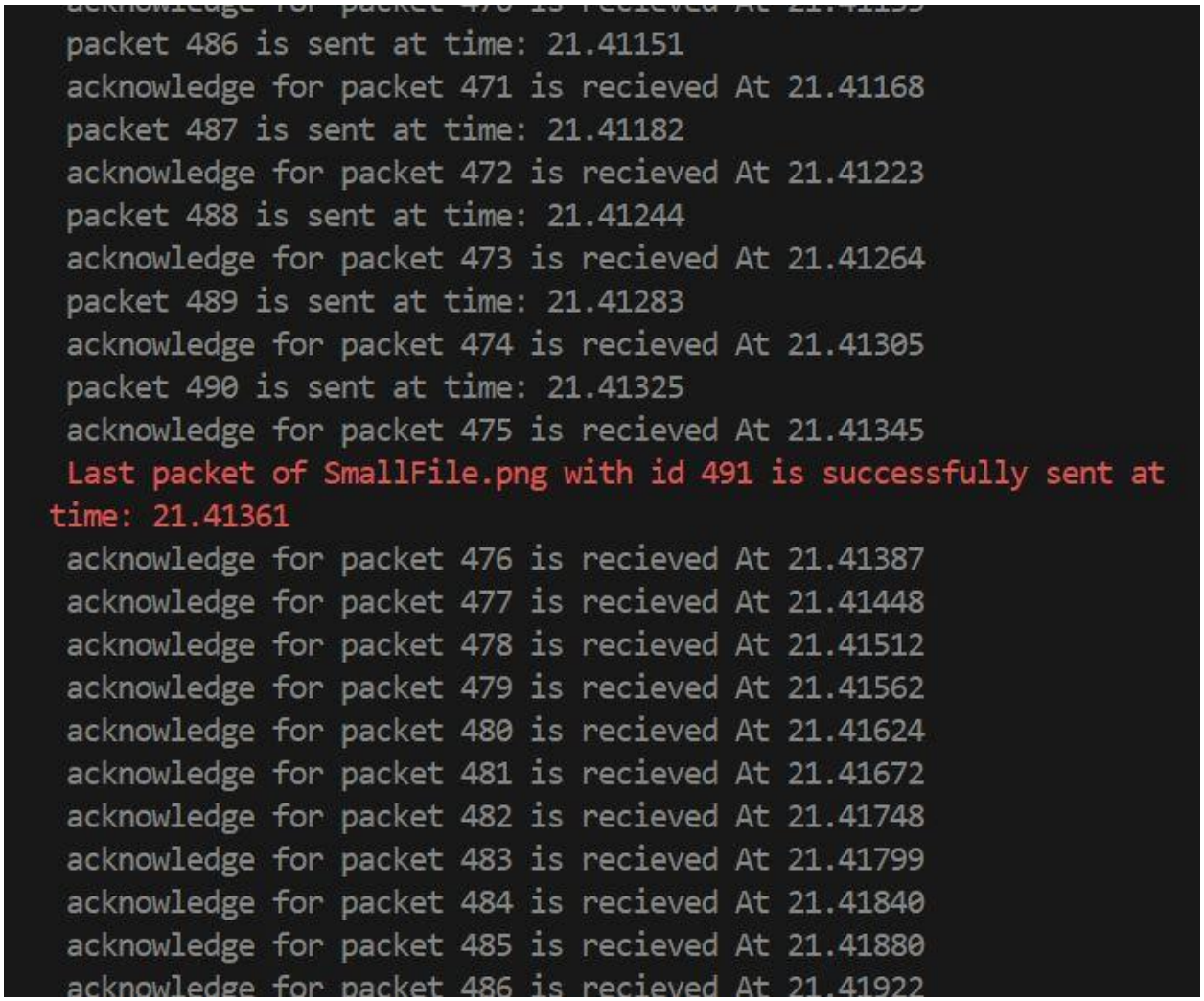
In our project we decreased the timeout interval (0.8) so the average transfer rate increases in order to enhance the solution. When we tried decreasing the timeout less than 0.8 we noticed that the loss rate increased.

The maximum segment size (MSS)

It is responsible for the size of each sent packet. If we increase the MSS, then we can improve the throughput due to reducing the overhead of packet headers and trailers. But, a larger MSS also increases the risk of packet loss, as larger packets are more likely to be corrupted or dropped due to errors in the network. So, the MSS should be selected according to the maximum transmission unit of the network as well as the error rate of the network.

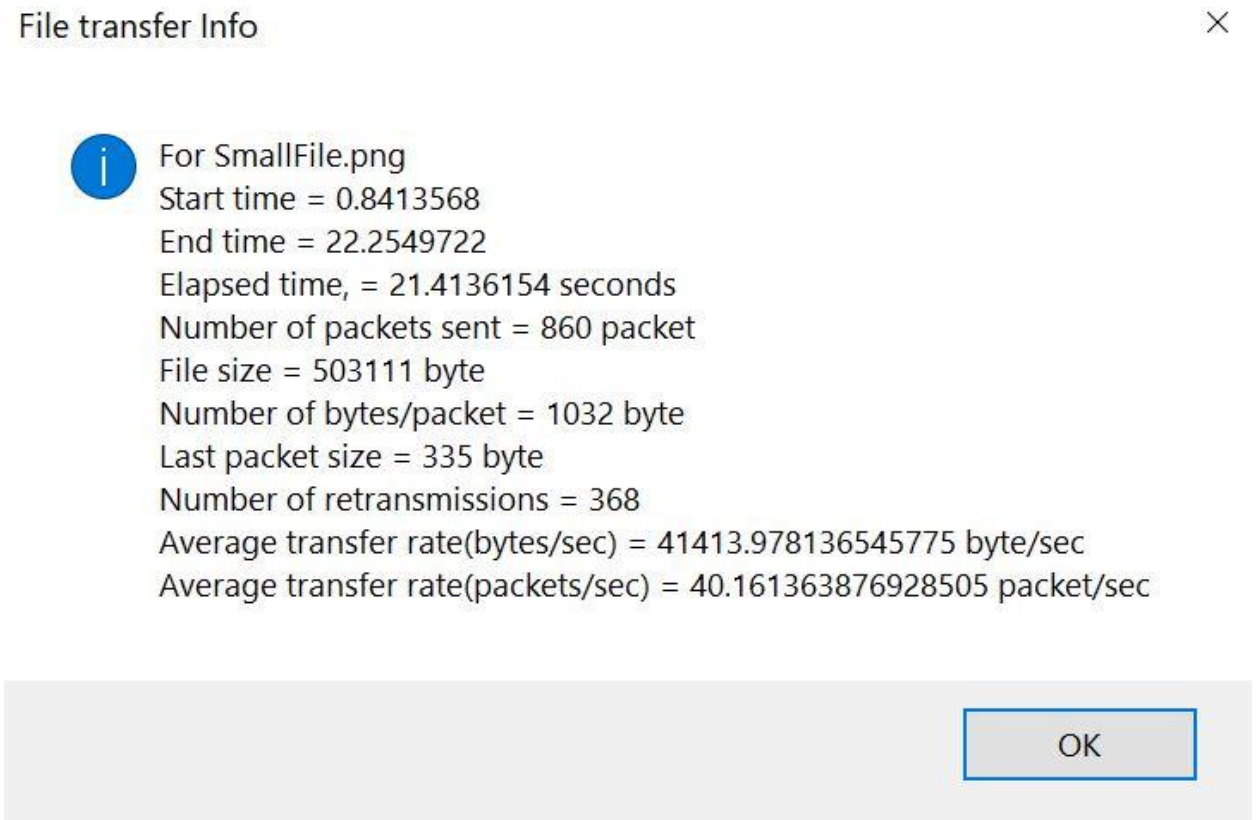
Sender:

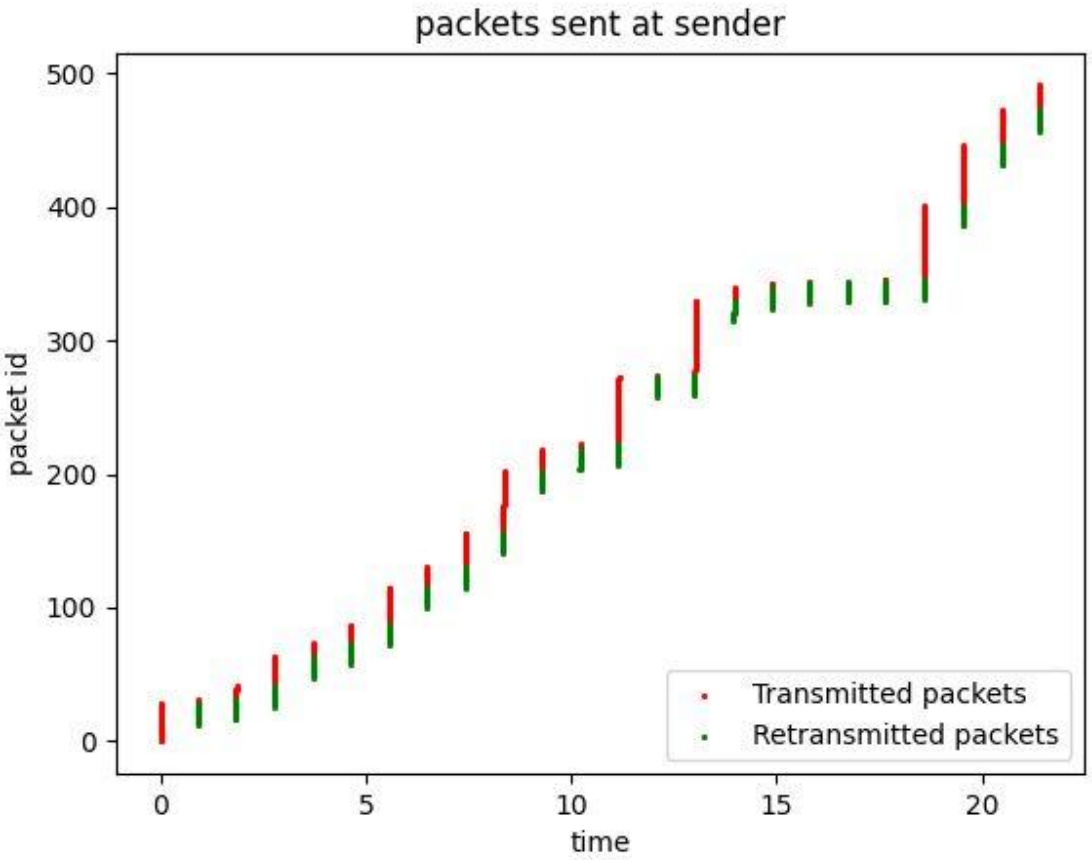
Output screenshots:



```
 acknowledge for packet 385 is recieved At 18.60103
 acknowledge for packet 385 is recieved At 18.60175
 acknowledge for packet 385 is recieved At 18.60231
 acknowledge for packet 385 is recieved At 18.60285
 acknowledge for packet 385 is recieved At 18.60330
 acknowledge for packet 385 is recieved At 18.60373
 acknowledge for packet 385 is recieved At 18.60433
timed out i At 19.51844
 packet 386 is sent at time: 19.51915
 packet 387 is sent at time: 19.51951
 packet 388 is sent at time: 19.51977
 packet 389 is sent at time: 19.52005
 packet 390 is sent at time: 19.52034
```

Statistics and graphs:





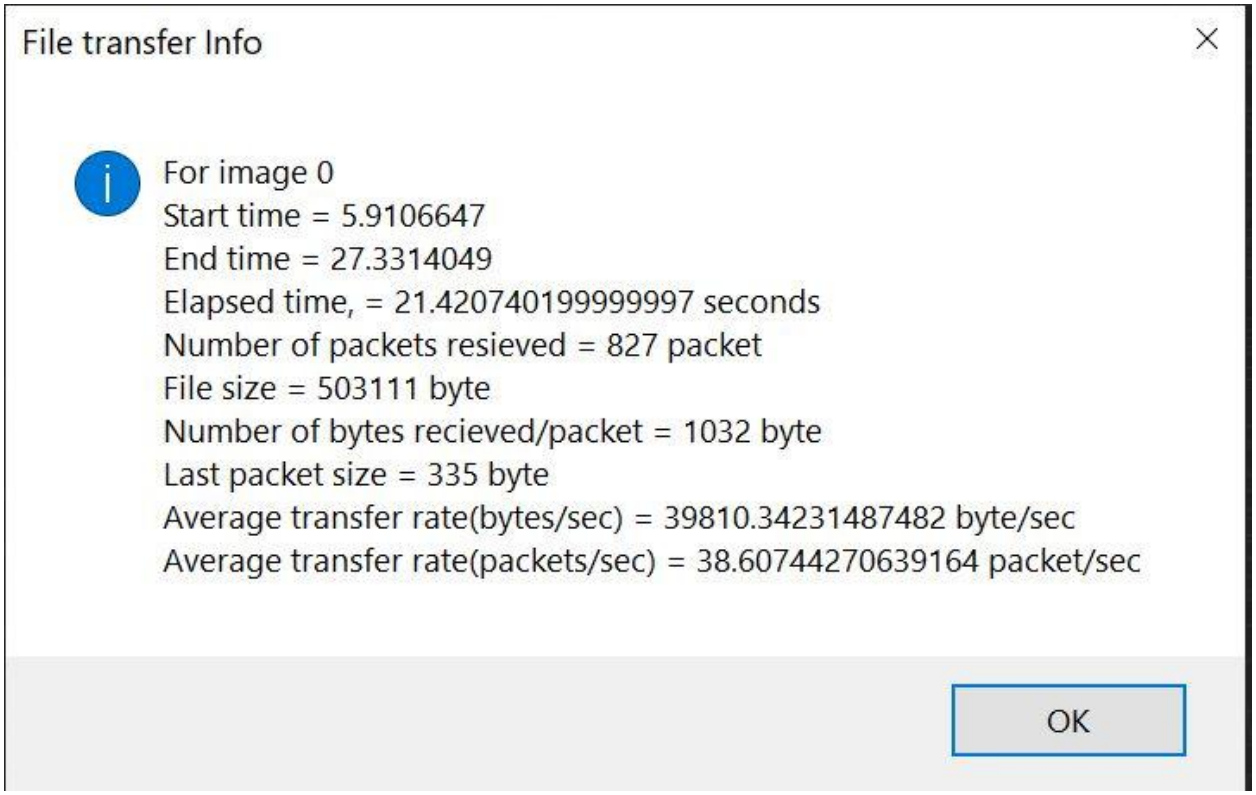
Receiver:

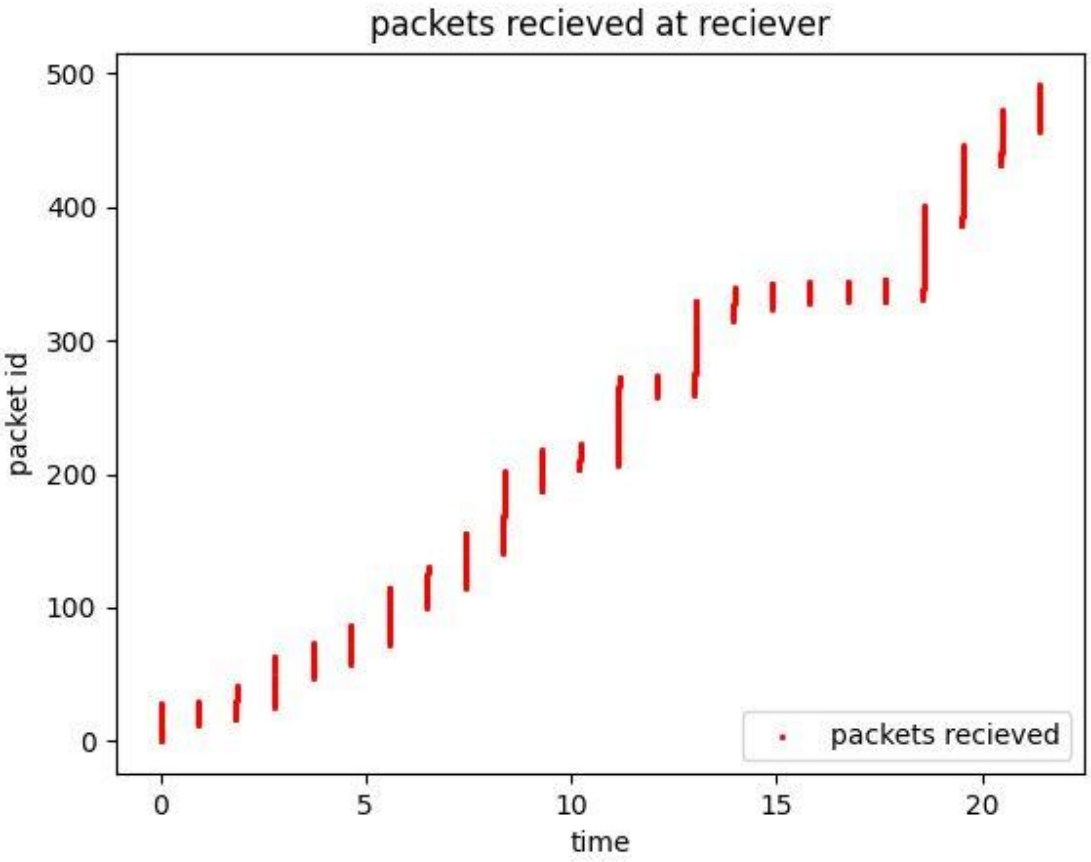
Output screenshots

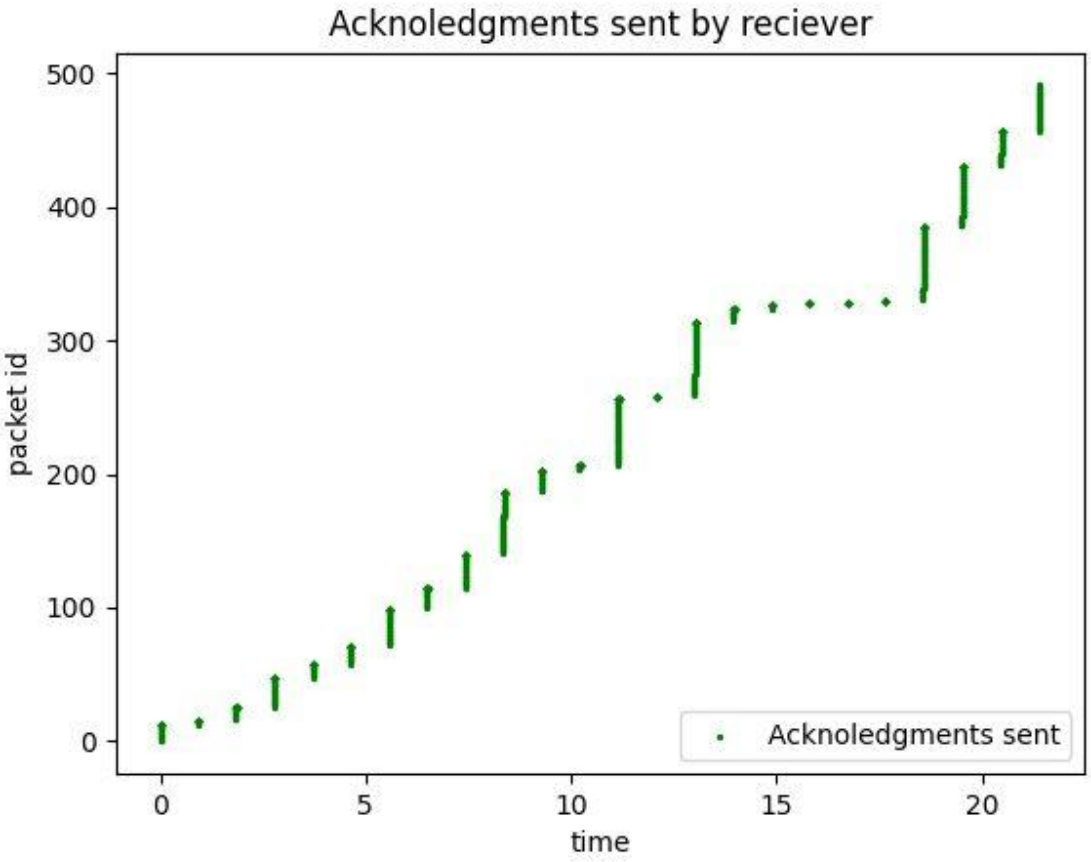
```
Sending Acknowledgment for packet 484 At 21.41715
 A Packet is recieved At 21.41732
It is An expected packet with ID =  485
Sending Acknowledgment for packet 485 At 21.41754
 A Packet is recieved At 21.41773
It is An expected packet with ID =  486
Sending Acknowledgment for packet 486 At 21.41797
 A Packet is recieved At 21.41814
It is An expected packet with ID =  487
Sending Acknowledgment for packet 487 At 21.41836
 A Packet is recieved At 21.41856
It is An expected packet with ID =  488
Sending Acknowledgment for packet 488 At 21.41897
 A Packet is recieved At 21.41921
It is An expected packet with ID =  489
Sending Acknowledgment for packet 489 At 21.41954
 A Packet is recieved At 21.41972
It is An expected packet with ID =  490
Sending Acknowledgment for packet 490 At 21.42001
 A Packet is recieved At 21.42026
It is An expected packet with ID =  491
Sending Acknowledgment for packet 491 At 21.42050
tarawa
 File image 0 reception is completed At 21.42074 with packet id 4
91
```

```
 A Packet is recieved At 18.59918
This is An unexpected packet with ID = 393
Sending last correct Acknowledgment At 18.59940
 A Packet is recieved At 18.59955
This is An unexpected packet with ID = 394
Sending last correct Acknowledgment At 18.59976
 dropped packet here!
 A Packet is recieved At 18.60012
This is An unexpected packet with ID = 396
Sending last correct Acknowledgment At 18.60043
 A Packet is recieved At 18.60067
This is An unexpected packet with ID = 397
Sending last correct Acknowledgment At 18.60101
```

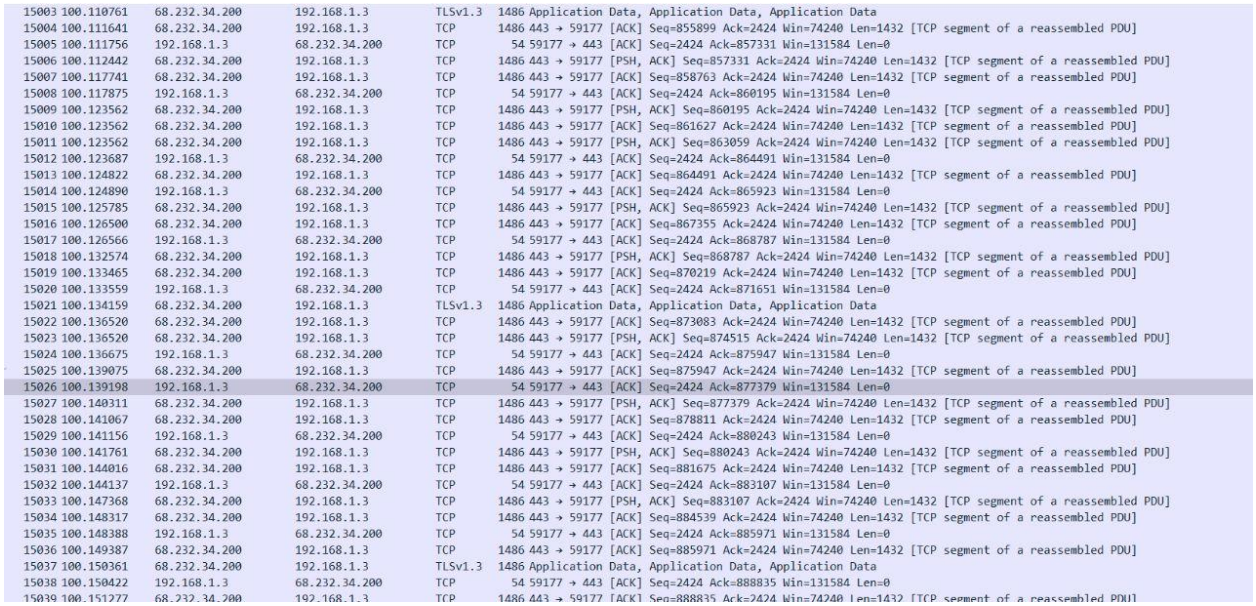
Statistics and graphs




packets recieved at reciever
packet id
time
packets recieved
0
100
200
300
400
500
0
5
10
15
20



Wireshark Simulation:



15003	100.110761	68.232.34.200	192.168.1.3	TLSv1.3	1486 Application Data, Application Data, Application Data
15004	100.111641	68.232.34.200	192.168.1.3	TCP	1486 443 → 59177 [ACK] Seq=855899 Ack=2424 Win=74240 Len=1432 [TCP segment of a reassembled PDU]
15005	100.111756	192.168.1.3	68.232.34.200	TCP	54 59177 → 443 [ACK] Seq=2424 Ack=857331 Win=131584 Len=0
15006	100.112442	68.232.34.200	192.168.1.3	TCP	1486 443 → 59177 [PSH, ACK] Seq=857331 Ack=2424 Win=74240 Len=1432 [TCP segment of a reassembled PDU]
15007	100.117741	68.232.34.200	192.168.1.3	TCP	1486 443 → 59177 [ACK] Seq=858763 Ack=2424 Win=74240 Len=1432 [TCP segment of a reassembled PDU]
15008	100.117875	192.168.1.3	68.232.34.200	TCP	54 59177 → 443 [ACK] Seq=2424 Ack=860195 Win=131584 Len=0
15009	100.123562	68.232.34.200	192.168.1.3	TCP	1486 443 → 59177 [PSH, ACK] Seq=860195 Ack=2424 Win=74240 Len=1432 [TCP segment of a reassembled PDU]
15010	100.123562	68.232.34.200	192.168.1.3	TCP	1486 443 → 59177 [ACK] Seq=861627 Ack=2424 Win=74240 Len=1432 [TCP segment of a reassembled PDU]
15011	100.123562	68.232.34.200	192.168.1.3	TCP	1486 443 → 59177 [PSH, ACK] Seq=863059 Ack=2424 Win=74240 Len=1432 [TCP segment of a reassembled PDU]
15012	100.123687	192.168.1.3	68.232.34.200	TCP	54 59177 → 443 [ACK] Seq=2424 Ack=864491 Win=131584 Len=0
15013	100.124822	68.232.34.200	192.168.1.3	TCP	1486 443 → 59177 [ACK] Seq=864491 Ack=2424 Win=74240 Len=1432 [TCP segment of a reassembled PDU]
15014	100.124890	192.168.1.3	68.232.34.200	TCP	54 59177 → 443 [ACK] Seq=2424 Ack=865923 Win=131584 Len=0
15015	100.125785	68.232.34.200	192.168.1.3	TCP	1486 443 → 59177 [PSH, ACK] Seq=865923 Ack=2424 Win=74240 Len=1432 [TCP segment of a reassembled PDU]
15016	100.126500	68.232.34.200	192.168.1.3	TCP	1486 443 → 59177 [ACK] Seq=867355 Ack=2424 Win=74240 Len=1432 [TCP segment of a reassembled PDU]
15017	100.126566	192.168.1.3	68.232.34.200	TCP	54 59177 → 443 [ACK] Seq=2424 Ack=868787 Win=131584 Len=0
15018	100.132574	68.232.34.200	192.168.1.3	TCP	1486 443 → 59177 [PSH, ACK] Seq=868787 Ack=2424 Win=74240 Len=1432 [TCP segment of a reassembled PDU]
15019	100.133465	68.232.34.200	192.168.1.3	TCP	1486 443 → 59177 [ACK] Seq=870219 Ack=2424 Win=74240 Len=1432 [TCP segment of a reassembled PDU]
15020	100.133559	192.168.1.3	68.232.34.200	TCP	54 59177 → 443 [ACK] Seq=2424 Ack=871651 Win=131584 Len=0
15021	100.134159	68.232.34.200	192.168.1.3	TLSv1.3	1486 Application Data, Application Data, Application Data
15022	100.136520	68.232.34.200	192.168.1.3	TCP	1486 443 → 59177 [ACK] Seq=873083 Ack=2424 Win=74240 Len=1432 [TCP segment of a reassembled PDU]
15023	100.136520	68.232.34.200	192.168.1.3	TCP	1486 443 → 59177 [PSH, ACK] Seq=874515 Ack=2424 Win=74240 Len=1432 [TCP segment of a reassembled PDU]
15024	100.136675	192.168.1.3	68.232.34.200	TCP	54 59177 → 443 [ACK] Seq=2424 Ack=875947 Win=131584 Len=0
15025	100.139075	68.232.34.200	192.168.1.3	TCP	1486 443 → 59177 [ACK] Seq=875947 Ack=2424 Win=74240 Len=1432 [TCP segment of a reassembled PDU]
15026	100.139198	192.168.1.3	68.232.34.200	TCP	54 59177 → 443 [ACK] Seq=2424 Ack=877379 Win=131584 Len=0
15027	100.140311	68.232.34.200	192.168.1.3	TCP	1486 443 → 59177 [PSH, ACK] Seq=877379 Ack=2424 Win=74240 Len=1432 [TCP segment of a reassembled PDU]
15028	100.141067	68.232.34.200	192.168.1.3	TCP	1486 443 → 59177 [ACK] Seq=878811 Ack=2424 Win=74240 Len=1432 [TCP segment of a reassembled PDU]
15029	100.141156	192.168.1.3	68.232.34.200	TCP	54 59177 → 443 [ACK] Seq=2424 Ack=880243 Win=131584 Len=0
15030	100.141761	68.232.34.200	192.168.1.3	TCP	1486 443 → 59177 [PSH, ACK] Seq=880243 Ack=2424 Win=74240 Len=1432 [TCP segment of a reassembled PDU]
15031	100.144016	68.232.34.200	192.168.1.3	TCP	1486 443 → 59177 [ACK] Seq=881675 Ack=2424 Win=74240 Len=1432 [TCP segment of a reassembled PDU]
15032	100.144137	192.168.1.3	68.232.34.200	TCP	54 59177 → 443 [ACK] Seq=2424 Ack=883107 Win=131584 Len=0
15033	100.147368	68.232.34.200	192.168.1.3	TCP	1486 443 → 59177 [PSH, ACK] Seq=883107 Ack=2424 Win=74240 Len=1432 [TCP segment of a reassembled PDU]
15034	100.148317	68.232.34.200	192.168.1.3	TCP	1486 443 → 59177 [ACK] Seq=884539 Ack=2424 Win=74240 Len=1432 [TCP segment of a reassembled PDU]
15035	100.148388	192.168.1.3	68.232.34.200	TCP	54 59177 → 443 [ACK] Seq=2424 Ack=885971 Win=131584 Len=0
15036	100.149387	68.232.34.200	192.168.1.3	TCP	1486 443 → 59177 [ACK] Seq=885971 Ack=2424 Win=74240 Len=1432 [TCP segment of a reassembled PDU]
15037	100.150361	68.232.34.200	192.168.1.3	TLSv1.3	1486 Application Data, Application Data, Application Data
15038	100.150422	192.168.1.3	68.232.34.200	TCP	54 59177 → 443 [ACK] Seq=2424 Ack=888835 Win=131584 Len=0
15039	100.151277	68.232.34.200	192.168.1.3	TCP	1486 443 → 59177 [ACK] Seq=888835 Ack=2424 Win=74240 Len=1432 [TCP segment of a reassembled PDU]

Legal and social responsibility:

In Egypt, law No. 175 of 2018 regarding Anti-Cyber and information technology crimes:

Article 13 -regards encroaching on the security of information networks- implies that anyone who unduly uses any means of information technology in any communication services shall be punished by imprisonment for no less than three months and/or a fine of no less than 10,000 Egyptian Pounds and no more than 50,000 Egyptian Pounds.

- Article 16 -regards unlawful sniffing- implies that anyone who unduly intercepts any information or data shall be punished by imprisonment for no less than one year and/or a fine ranging from 50,000 to 250,000 Egyptian Pound.

Internationally, there are many laws. Such as the United Nations Charter and customary international law, apply to cyber attacks. The principles of sovereignty,

non-intervention, and the prohibition of the use of force are relevant in cyberspace. The Tallinn Manual, commissioned by the NATO Cooperative Cyber Defence Centre of Excellence, provides guidance on how international law, including the law of armed conflict, applies to cyber operations. The Council of Europe's Convention on Cybercrime, also known as the Budapest Convention, is an international treaty that aims to harmonize national laws and improve cooperation in investigating cybercrimes, including cyber attacks. There are also other initiatives and organizations like ASEAN Cybersecurity Cooperation Strategy, the African Union Convention on Cyber Security and Personal Data Protection, and the Organization of American States (OAS) Cybersecurity Program.

Social impact:

Cyber attacks can cause people’s data to be vulnerable to attackers. Data nowadays have become crucial in everyday life. Bank accounts data, credit cards, log in credentials, addresses etc.. Data is worth a lot of money. So, it is important to be ready for cyber attacks with proper defending methods.

Attacks & suggested mitigation methods

Malicious attacks are a serious threat to network communication. In this project, we were asked to modify the GBN sender in a way that exploits any weakness in the protocol's specification to maliciously disrupt the receiver. There are many kinds of attacks such as : packet injection, eavesdropping, and denial of service.

Possible attack (DoS)

Denial of Service (DoS) attack is a conceivable approach in which the attacker floods the network with a large number of packets to overwhelm and disrupt the receiver. In this scenario, the attacker can take advantage of the GBN protocol's

sliding window mechanism to send a large number of packets that exceed the window size of the receiver, leading the receiver to drop and retransmit packets. As a result, the receiver will be overburdened and unable to handle legitimate packets, resulting in a denial of service.

DoS mitigation

To mitigate this attack, we can install a packet rate limiter at the sender's end, limiting the number of packets transmitted per unit time. This prevents the attacker from delivering an excessive number of packets, which would overload the receiver. Furthermore, at the receiver's end, we can implement a packet filtering mechanism that drops packets that exceed a certain threshold, preventing the receiver from becoming overwhelmed.

Possible protocol updates for a more secure version

- **Checksum:**

In order to assure the integrity of the data being communicated, use a checksum technique. This prevents intruders from messing with the data and threatening its integrity.

- **Authentication mechanism:**

Install an authentication mechanism to validate the sender and receiver's identities, preventing attackers from impersonating authorized users and conducting attacks.

- **Congestion control mechanism:**

Introduce a congestion control mechanism that limits the number of packets sent based on the current state of the network. This keeps the network from getting overburdened and ensures that authentic packets are delivered.

- **Sequence Number Randomization:**

For each packet sent in the GBN protocol, the sequence number is increased by one. As a result, an attacker can easily guess the next sequence number and inject

bogus packets into the stream. To avoid this, we can randomize the sequence number so that an attacker cannot guess the next sequence number.

- **Access Control:**

It is critical to protect the network and the GBN protocol from unauthorized access. Access control technologies like user authentication, firewalls, and intrusion detection systems can help achieve this.

Our attempt to implement a more secure version

We have tried to implement Secure Sockets Layer (SSL) which is a security protocol that provides a secure communication channel between two machines over the internet. Moreover, we tried to implement a checksum function using the CRC algorithm.