



Go-Back-N Protocol

Implementation in Python

Deepanshu Jindal 2016CS10312
Ujjwal Gupta 2016CS10087



Implementation details

We used four classes namely:

- `Packet()`
- `NetworkLayer()`
- `Frame()`
- `DataLinkLayer()`



Implementation details

Packet

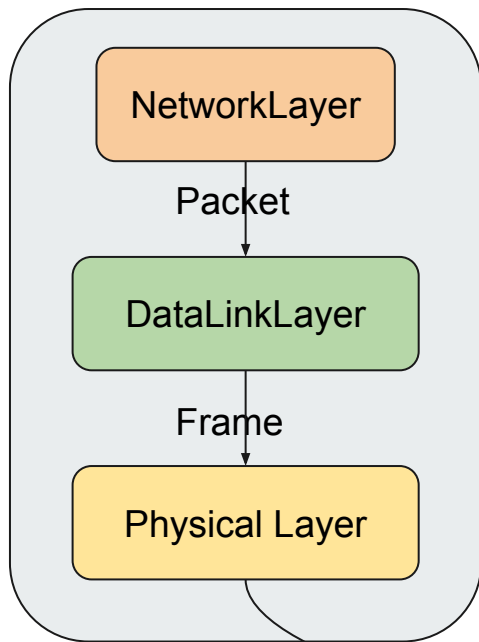
- Sequence Number
- Origin Host ID
- Payload : String of random length

Frame

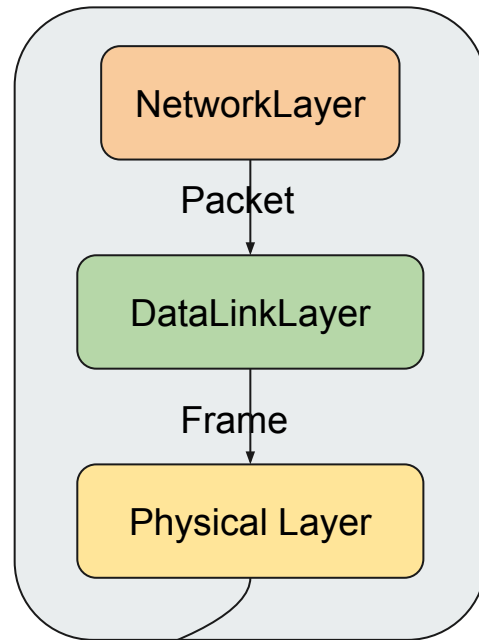
- Sequence Number
- Ack Number
- Enclosed Packet
- Checksum





Host 1





Host 2



- 
- Network Layer generates and receives packets and adds the log of packets in separate files along with timestamps.
 - The packet generation rate is controlled by `run_network_layer()` which switches the `NETWORK_LAYER_READY` variable with time.

- 
- Network Layer generates and receives packets and adds the log of packets in separate files along with timestamps.
 - The packet generation rate is controlled by `run_network_layer()` which switches the `NETWORK_LAYER_READY` variable with time.
 - Physical layer sends and receives packets to/from other host by encapsulating them into a datagram and transmitting via sockets.

- 
- Network Layer generates and receives packets and adds the log of packets in separate files along with timestamps.
 - The packet generation rate is controlled by `run_network_layer()` which switches the `NETWORK_LAYER_READY` variable with time.
 - Physical layer sends and receives packets to/from other host by encapsulating them into a frame and transmitting via sockets.
 - The network and physical layers are run through separate threads, apart from the main thread that runs data link layer.

- 
- Various events like:
 - Network Layer Ready
 - Frame Arrival
 - Checksum errors
 - Data and Ack management
 - Timeout

are handled by the Data Link Layer in function `GBN_protocol()`

- Network layer is enabled and disabled by this function only



How to run the code

- Setup mininet topology using the script `./topology_setup.sh`
- Open xterm for the hosts using `"xterm h1 h2"`
- On host1 run `"python GBN.py 1 <host1 ip> <host2 ip>"`
- On host2 run `"python GBN.py 2 <host2 ip> <host1 ip>"`
- `-v` [verbose flag] can be given to python scripts to print out the packets sent and received by network layer in a file



How to run the code

The code can be run directly on the system as well without using mininet. This gives much better performance.

- Open two terminals
- On host1 run `python GBN.py 1 <host1 ip> <host2 ip>`
- On host2 run `python GBN.py 2 <host2 ip> <host1 ip>`
- `-v` [verbose flag] can be given to python scripts to print out the packets sent and received by network layer in a file



How to run the code

- To modify link parameters modify the arguments given `add_link()` function `topology_setup.py`
- To modify window size, change the first argument given to constructor for `DataLinkLayer()`



How to run the code

- All the events on a frame are printed on console which can be redirected to suitable file for analysis
- A summary of transmission is printed after running the code for the given time limit
- Packets sent and received are printed along with timestamps in sent_packets and receive_packets files with host_id as suffix



End to end delay

- Components of end to end delay in real networks : -
 - Transmission Delay
 - Propagation Delay
 - Processing Delay
 - Queueing Delay



End to end delay

- In our simulation
 - Transmission delay comes from the link delay set while topology setup on mininet
 - Processing delay is emulated by the time consumed in encapsulating packets into frames and stringifying the packets



End to end delay

- In our simulation
 - Propagation delay is emulated by the inherent delay in transmitting packets using socket connection.
 - Queueing delay is emulated by the time a packet spends in the buffer before it is processed by the data link layer



Throughput

- Throughput is the rate at which packets are delivered successfully from source to destination within a given time period.
- It can be measured in terms of packets/frames transmitted/received per second.



Throughput

- In our simulation two factors govern the throughput:
 - Network Layer is constrained to send only 400 packets per second
(This places a cap on fresh transmissions generated)
 - Bandwidth of the link is restricted at 1 Mbps
(This places the cap on retransmissions done)



Drop rate

- Packet drop/loss occurs in networks when one or more packets travelling across a network fail to reach their destination.
- Reasons for packet loss :-
 - Network Congestion
 - Unstable channel
 - Inherent unreliability
- Drop rate refers to the ratio of packets dropped over the total packets transmitted by a host.



Drop rate

- In our simulation, this is governed by the link parameters defined during the topology_setup on mininet.
- We also tried setting it up by probabilistically dropping packet at the receiver's end however, both the methods gave similar results so we decided to go with the first method only due to its simplicity.



Data flow summaries with different parameters



Window Size = 5, Loss = 0.01, Delay = 5ms

Host 1 TRANSMISSION SUMMARY

Total Frames transmitted: 220
Frames retransmitted: 30
Packets sent: 190
Payload bytes sent: 23785

Host 1 RECEPTION SUMMARY

Frames received with checksum error: 0
Frames received with no error: 186
Packets received: 186
Payload bytes received: 23152

Host 2 TRANSMISSION SUMMARY

Total Frames transmitted: 228
Frames retransmitted: 34
Packets sent: 194
Payload bytes sent: 24121

Host 2 RECEPTION SUMMARY

Frames received with checksum error: 0
Frames received with no error: 190
Packets received: 190
Payload bytes received: 23785



Window Size = 7, Loss = 0.01, Delay = 5ms

Host 1 TRANSMISSION SUMMARY

Total Frames transmitted: 312
Frames retransmitted: 48
Packets sent: 264
Payload bytes sent: 30481

Host 1 RECEPTION SUMMARY

Frames received with checksum error: 0
Frames received with no error: 245
Packets received: 245
Payload bytes received: 28132

Host 2 TRANSMISSION SUMMARY

Total Frames transmitted: 327
Frames retransmitted: 59
Packets sent: 268
Payload bytes sent: 31155

Host 2 RECEPTION SUMMARY

Frames received with checksum error: 0
Frames received with no error: 264
Packets received: 264
Payload bytes received: 30481



Window Size = 9, Loss = 0.01, Delay = 5ms

Host 1 TRANSMISSION SUMMARY

Total Frames transmitted: 382
Frames retransmitted: 69
Packets sent: 313
Payload bytes sent: 37367

Host 1 RECEPTION SUMMARY

Frames received with checksum error: 0
Frames received with no error: 259
Packets received: 259
Payload bytes received: 31173

Host 2 TRANSMISSION SUMMARY

Total Frames transmitted: 329
Frames retransmitted: 64
Packets sent: 265
Payload bytes sent: 33382

Host 2 RECEPTION SUMMARY

Frames received with checksum error: 0
Frames received with no error: 313
Packets received: 313
Payload bytes received: 37367



Increasing Window Size

More aggressive transmissions

Higher throughput but high retransmissions as well

Overall higher successful transmissions



Window Size = 7, Loss = 0.05, Delay = 5ms

Host 1 TRANSMISSION SUMMARY

Total Frames transmitted: 232
Frames retransmitted: 88
Packets sent: 144
Payload bytes sent: 16695

Host 1 RECEPTION SUMMARY

Frames received with checksum error: 0
Frames received with no error: 110
Packets received: 110
Payload bytes received: 13930

Host 2 TRANSMISSION SUMMARY

Total Frames transmitted: 219
Frames retransmitted: 88
Packets sent: 131
Payload bytes sent: 16202

Host 2 RECEPTION SUMMARY

Frames received with checksum error: 0
Frames received with no error: 136
Packets received: 136
Payload bytes received: 15874



Window Size = 7, Loss = 0.1, Delay = 5ms

Host 1 TRANSMISSION SUMMARY

Total Frames transmitted: 214
Frames retransmitted: 154
Packets sent: 60
Payload bytes sent: 6677

Host 1 RECEPTION SUMMARY

Frames received with checksum error: 0
Frames received with no error: 74
Packets received: 74
Payload bytes received: 8935

Host 2 TRANSMISSION SUMMARY

Total Frames transmitted: 227
Frames retransmitted: 146
Packets sent: 81
Payload bytes sent: 9766

Host 2 RECEPTION SUMMARY

Frames received with checksum error: 0
Frames received with no error: 60
Packets received: 60
Payload bytes received: 6677



Increasing Error/Loss rate

Number of retransmissions required increases abruptly

Overall effective throughput decreases

At high loss rates (10%), the number of successful transmissions become extremely low



Window Size = 7, Loss = 0.01, Delay = 0ms

Host 1 TRANSMISSION SUMMARY

Total Frames transmitted: 325
Frames retransmitted: 44
Packets sent: 281
Payload bytes sent: 33356

Host 1 RECEPTION SUMMARY

Frames received with checksum error: 0
Frames received with no error: 263
Packets received: 263
Payload bytes received: 32075

Host 2 TRANSMISSION SUMMARY

Total Frames transmitted: 332
Frames retransmitted: 60
Packets sent: 272
Payload bytes sent: 32689

Host 2 RECEPTION SUMMARY

Frames received with checksum error: 0
Frames received with no error: 276
Packets received: 276
Payload bytes received: 32981



Window Size = 7, Loss = 0.01, Delay = 3ms

Host 1 TRANSMISSION SUMMARY

Total Frames transmitted: 320
Frames retransmitted: 46
Packets sent: 274
Payload bytes sent: 33046

Host 1 RECEPTION SUMMARY

Frames received with checksum error: 0
Frames received with no error: 258
Packets received: 258
Payload bytes received: 30075

Host 2 TRANSMISSION SUMMARY

Total Frames transmitted: 320
Frames retransmitted: 56
Packets sent: 264
Payload bytes sent: 31189

Host 2 RECEPTION SUMMARY

Frames received with checksum error: 0
Frames received with no error: 272
Packets received: 272
Payload bytes received: 32941



Increasing delay

Slight impact on performance

With increased delay number of transmissions fall down by marginal amounts and retransmissions increases

Overall successful transmissions are reduced slightly



Submission contents

- GBN.py is the main source code for the implementation
- topology_setup.sh and topology_setup.py are for mininet setup
- Log files for host1 and host2



Sources of Errors

- There are some discrepancies between the number of packets sent and the number of packets received by other host.
 - Since threads are timed out abruptly, it is possible some packets were left in buffer unprocessed.
 - Also setup time of sockets may incur some losses.
- Initially there are some timeouts and thereafter the flow stabilises due to setup time of sockets and differential running time of scripts on different terminals
- The packet generation rate has been fine tuned after some trial and error with different values which may differ slightly on different systems