Laxman Muthe

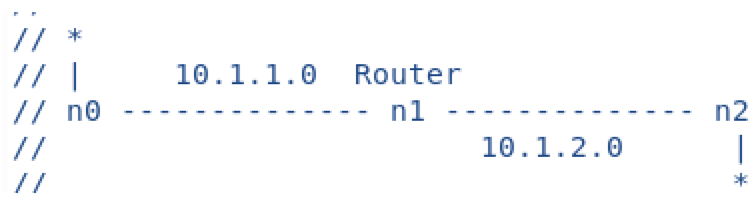
906602548

laxman@vt.edu

ECE/CS 5566 Project 3

This report presents the findings from a series of simulations that analyze the performance of the TCP/IP suite under varying packet loss rates. Using the ns-3 network simulator, we modeled a simple network to measure the impact of error detection and retransmission mechanisms on overall network throughput and latency.

1) Simulation Model



```
// *
// |      10.1.1.0  Router
// n0 -------------- n1 -------------- n2
//                        10.1.2.0      |
//                                      *
```

Our simulation model replicates a simple TCP/IP network consisting of three nodes connected in a topology as shown above. The nodes are connected as follows: n0 is connected to n1, which in turn is connected to n2.

Three nodes (n0, n1, n2) are created. n0 serves as the source of TCP packets, n1 functions as a router and n2 is the destination for the TCP packets.

The nodes are interconnected through P2P links. The first link connects n0 to n1 and the second link connects n1 to n2. These links are configured with a data rate of 10 Mbps and a delay of 10 ms according to requirements.

Two IP networks are defined: 10.1.1.1 for the n0-n1 link and 10.1.2.2 for the n1-n2 link. IP addresses are assigned accordingly to each interface on these links.

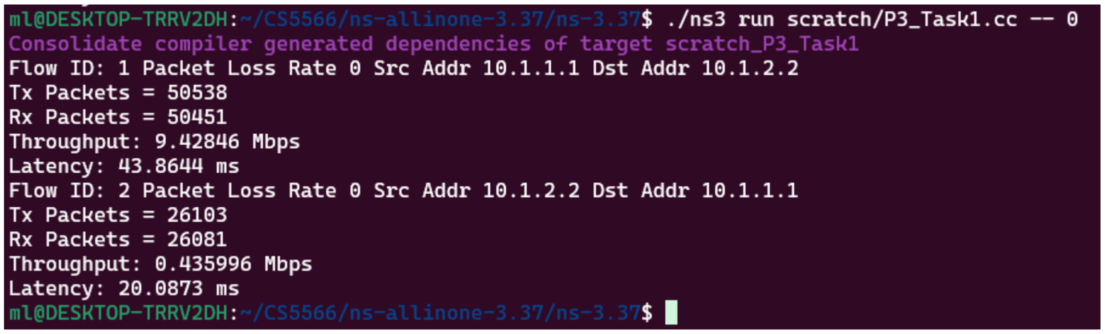
Global routing is enabled to automatically manage the routing tables for all nodes, ensuring that packets can be forwarded from the source to the destination across the network.

The source node (n0) generates TCP traffic directed at the destination node (n2) using a TcpSocketFactory, which continuously sends data starting after a warmup time of 5 seconds until the end of the simulation. The destination (n2) has a PacketSinkHelper to receive the TCP traffic.

An error model with error unit ERROR_UNIT_PACKET is applied to the receiving end of both P2P links, simulating packet loss with a specified rate. This introduces errors into the network, affecting packet delivery.

The simulation includes flow monitoring to track and report statistics on the TCP flow, such as packets transmitted (Tx), packets received (Rx), throughput and end-to-end latency. The simulation runs for a specified duration of 30 seconds(was later changed to 110 seconds) and it can be configured to output tracing information, such as pcap files, for detailed analysis. Also, a warmup time of 5 seconds is used for the stable setup.

Error rate is passed through command line argument



```
ml@DESKTOP-TRRV2DH:~/CS5566/ns-allinone-3.37/ns-3.37$ ./ns3 run scratch/P3_Task1.cc -- 0
Consolidate compiler generated dependencies of target scratch_P3_Task1
Flow ID: 1 Packet Loss Rate 0 Src Addr 10.1.1.1 Dst Addr 10.1.2.2
Tx Packets = 50538
Rx Packets = 50451
Throughput: 9.42846 Mbps
Latency: 43.8644 ms
Flow ID: 2 Packet Loss Rate 0 Src Addr 10.1.2.2 Dst Addr 10.1.1.1
Tx Packets = 26103
Rx Packets = 26081
Throughput: 0.435996 Mbps
Latency: 20.0873 ms
ml@DESKTOP-TRRV2DH:~/CS5566/ns-allinone-3.37/ns-3.37$
```

Above is the output for simulation with error rate 0 for transmission of packets from source node n0 to destination node n1. Similarly, many pilot runs were taken to understand the latency and throughput for different P/error rates.

2) Experiment Design: Parameters, Factors, and Metrics

ErrorRate: The variable, ranging from 0 (no loss) to 0.2 (20% loss), representing the probability of packet loss on a link. Based on simulation runs for each P/error rate, it was found that transmitted packets were decreasing as the error rate was increasing. Came to a good threshold of 0.2 to have some traffic for analysis.

In networking, port numbers are used to identify different services or processes running on a host. Ports numbered from 0 to 1023 are typically reserved for well-known services and port 9 is commonly associated with the "discard" service so we went with port 9.

DataRate and Delay: Held constant as part of the controlled experiment parameters as given in requirements.

Throughput: Calculated as the number of successfully received bits at the destination per unit of time, measured in Mbps.

Latency: The average time taken for packets to travel from the source node to the destination node, measured in milliseconds.

FlowMonitor was utilized to track packet flow, calculate throughput and measure latency across the network.

Used ReceiveErrorModel which is used to simulate errors during packet transmission, which can occur due to various factors such as signal degradation, noise and interference in communication channels.

ERROR_UNIT_PACKET, it tells the error model to apply the error rate to packets. This means that if a packet is chosen to have an error, the entire packet is considered erroneous and will be dropped.

Created a “simulation_results.csv” file and loaded it using C++ ofstream is append mode to store the data collected and keep on appending the data to it for each error rate.

Used Simulator::Schedule function which is used to schedule events to take place at a specific time in the simulation.

Seconds(t): This is the time at which the event is scheduled to occur. The t variable is the specific time in seconds. If t is 5.0, the event is scheduled to happen 5 seconds into the simulation.

[classifier, monitor, t, &outFd, errorRate]: This is a lambda function capture list. It captures the necessary variables by value or reference so they can be used within the lambda function. Here, classifier and monitor are captured by value, t and errorRate are captured by value and outFd which is used to open file, is captured by reference.

RunSimulation(classifier, monitor, t, outFd, errorRate): This is the function that is called when the event is triggered. It will run the simulation using the captured variables.

3) Simulation Runs

Our simulation experiments were designed to analyze the performance impact of packet loss rate (error rate) on TCP/IP networking, specifically looking at throughput and end-to-end latency as key performance metrics.

Each simulation was run by passing the error rate as command line argument parameterized by the packet loss rate, which was varied to observe the effect on the network performance. The ns-3 network simulator was employed to carry out these experiments, using a simple point-to-point topology to model the network.

To address the transient phase of the simulations, a warm-up time of 5 seconds was introduced before starting the traffic from flows. This allowed the simulation to stabilize and the TCP algorithms to adjust to the steady state before measurements were taken.

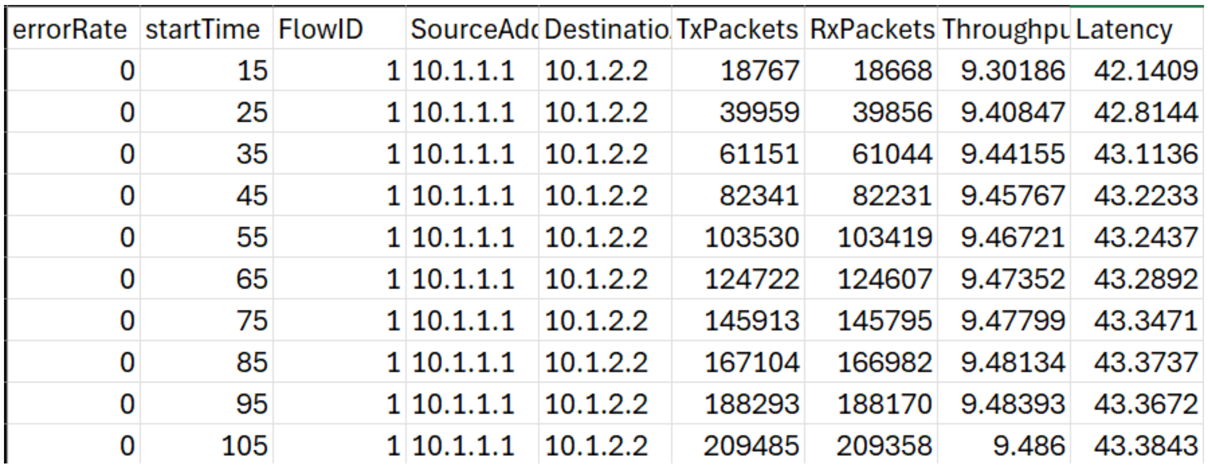
The simulation was run for 110 seconds, which provided enough time to observe the system's behavior under steady-state conditions. The length of the simulation was determined based on pilot runs that identified the time it took for the metrics of interest to stabilize. These pilot runs were crucial for establishing a simulation duration that was neither too short (potentially missing the steady-state behavior) nor too long (unnecessarily consuming computational resources). It was also based on the batch size to get a size of 10.

Before the main set of simulations, a series of pilot runs with varying error rates were conducted. Initially experimented with a random error rate where we found that for higher error rates very few packets to 0 were observed so reduced the error rates to a max of 0.2 where some traffic was observed instead of 0 or single digit packets. Also, starting the simulation at 0 seconds resulted in lesser packets than starting after a 5 second warm up time. These runs helped to identify appropriate ranges for the error rate, how to effectively remove start-up transients and the required length of the simulation to ensure accurate and reliable data collection. For example, pilot runs with low (0.01) and high (0.2) error rates gave us insights into the system's behavior, which informed the decision to set the simulation time to 110 seconds and the warm-up period to 5 seconds.

4) Data Analysis and Results

For the performance evaluation of the TCP/IP network under different error conditions, the batch means method was employed. This technique involves segmenting the simulation output data into batches and treating the mean of each batch as a sample to estimate the population mean and variance. Our simulation generated samples every 10 seconds, post a 5-second warm-up period, to ensure independence between samples.

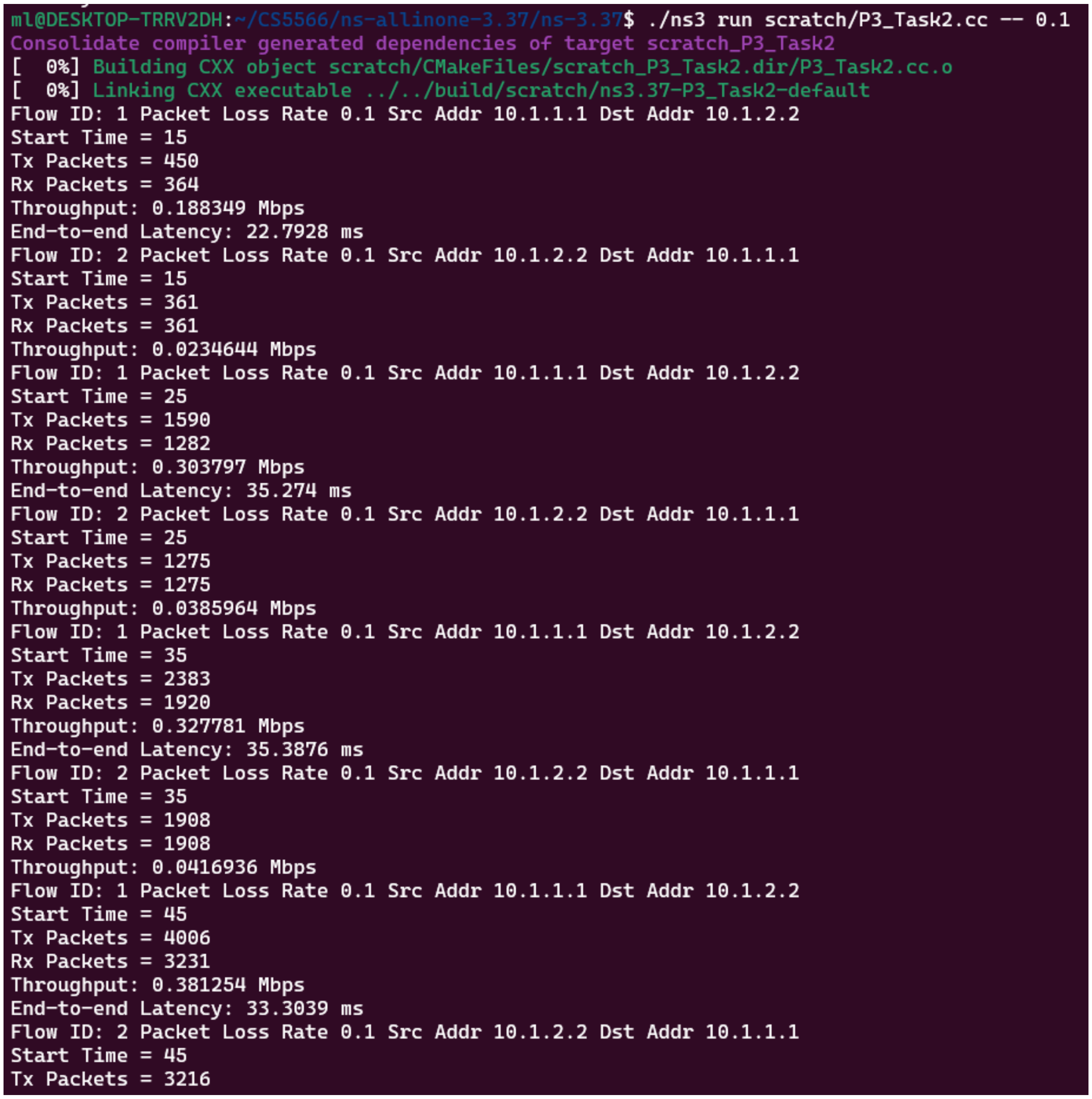
Simulation was done by passing each error rate one by one via command line arguments and performance metrics like start time, throughput, latency were appended to “simulation_results.csv” file.



errorRate	startTime	FlowID	SourceAdc	Destinatio	TxPackets	RxPackets	Throughpu	Latency
0	15	1	10.1.1.1	10.1.2.2	18767	18668	9.30186	42.1409
0	25	1	10.1.1.1	10.1.2.2	39959	39856	9.40847	42.8144
0	35	1	10.1.1.1	10.1.2.2	61151	61044	9.44155	43.1136
0	45	1	10.1.1.1	10.1.2.2	82341	82231	9.45767	43.2233
0	55	1	10.1.1.1	10.1.2.2	103530	103419	9.46721	43.2437
0	65	1	10.1.1.1	10.1.2.2	124722	124607	9.47352	43.2892
0	75	1	10.1.1.1	10.1.2.2	145913	145795	9.47799	43.3471
0	85	1	10.1.1.1	10.1.2.2	167104	166982	9.48134	43.3737
0	95	1	10.1.1.1	10.1.2.2	188293	188170	9.48393	43.3672
0	105	1	10.1.1.1	10.1.2.2	209485	209358	9.486	43.3843

For example, for the error rate, all the traffic information is captured flowing from 10.1.1.1(n0) to 10.1.2.2(n1). This is considered as a batch pass for P=0 which has batch size of 10 with throughput and latency as sample means. In a similar way, this process is carried out for all other error rates and stored in the same file.

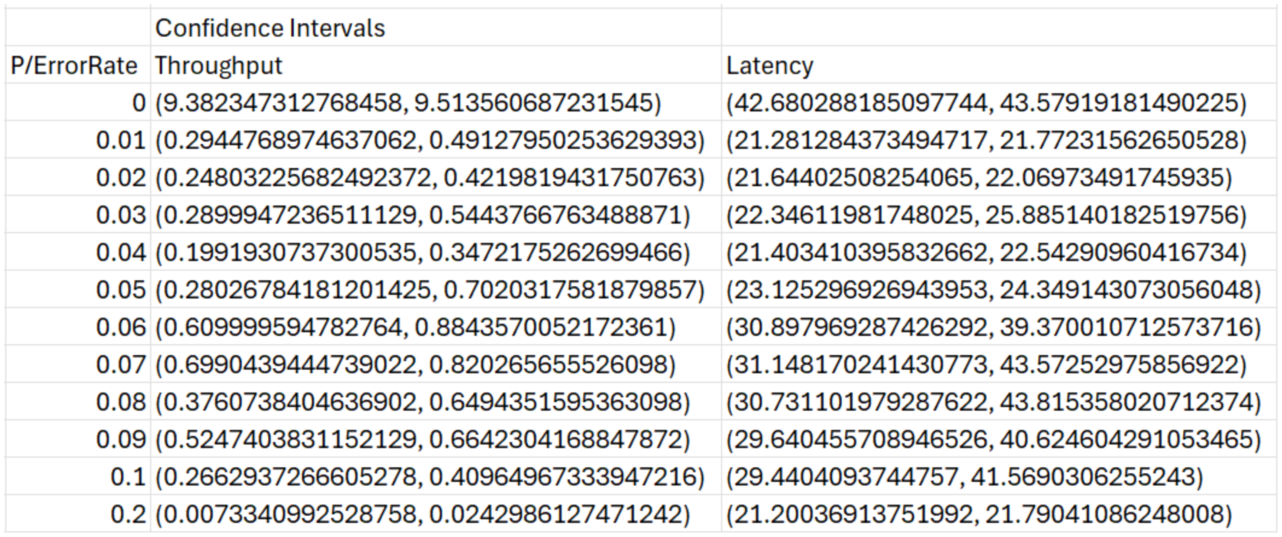
This data is now loaded in a python program that reads using pandas dataframe and calculates the confidence interval for the batch with error rate 0.



```
ml@DESKTOP-TRRV2DH:~/CS5566/ns-allinone-3.37/ns-3.37$ ./ns3 run scratch/P3_Task2.cc -- 0.1
Consolidate compiler generated dependencies of target scratch_P3_Task2
[  0%] Building CXX object scratch/CMakeFiles/scratch_P3_Task2.dir/P3_Task2.cc.o
[  0%] Linking CXX executable ../../build/scratch/ns3.37-P3_Task2-default
Flow ID: 1 Packet Loss Rate 0.1 Src Addr 10.1.1.1 Dst Addr 10.1.2.2
Start Time = 15
Tx Packets = 450
Rx Packets = 364
Throughput: 0.188349 Mbps
End-to-end Latency: 22.7928 ms
Flow ID: 2 Packet Loss Rate 0.1 Src Addr 10.1.2.2 Dst Addr 10.1.1.1
Start Time = 15
Tx Packets = 361
Rx Packets = 361
Throughput: 0.0234644 Mbps
Flow ID: 1 Packet Loss Rate 0.1 Src Addr 10.1.1.1 Dst Addr 10.1.2.2
Start Time = 25
Tx Packets = 1590
Rx Packets = 1282
Throughput: 0.303797 Mbps
End-to-end Latency: 35.274 ms
Flow ID: 2 Packet Loss Rate 0.1 Src Addr 10.1.2.2 Dst Addr 10.1.1.1
Start Time = 25
Tx Packets = 1275
Rx Packets = 1275
Throughput: 0.0385964 Mbps
Flow ID: 1 Packet Loss Rate 0.1 Src Addr 10.1.1.1 Dst Addr 10.1.2.2
Start Time = 35
Tx Packets = 2383
Rx Packets = 1920
Throughput: 0.327781 Mbps
End-to-end Latency: 35.3876 ms
Flow ID: 2 Packet Loss Rate 0.1 Src Addr 10.1.2.2 Dst Addr 10.1.1.1
Start Time = 35
Tx Packets = 1908
Rx Packets = 1908
Throughput: 0.0416936 Mbps
Flow ID: 1 Packet Loss Rate 0.1 Src Addr 10.1.1.1 Dst Addr 10.1.2.2
Start Time = 45
Tx Packets = 4006
Rx Packets = 3231
Throughput: 0.381254 Mbps
End-to-end Latency: 33.3039 ms
Flow ID: 2 Packet Loss Rate 0.1 Src Addr 10.1.2.2 Dst Addr 10.1.1.1
Start Time = 45
Tx Packets = 3216
```

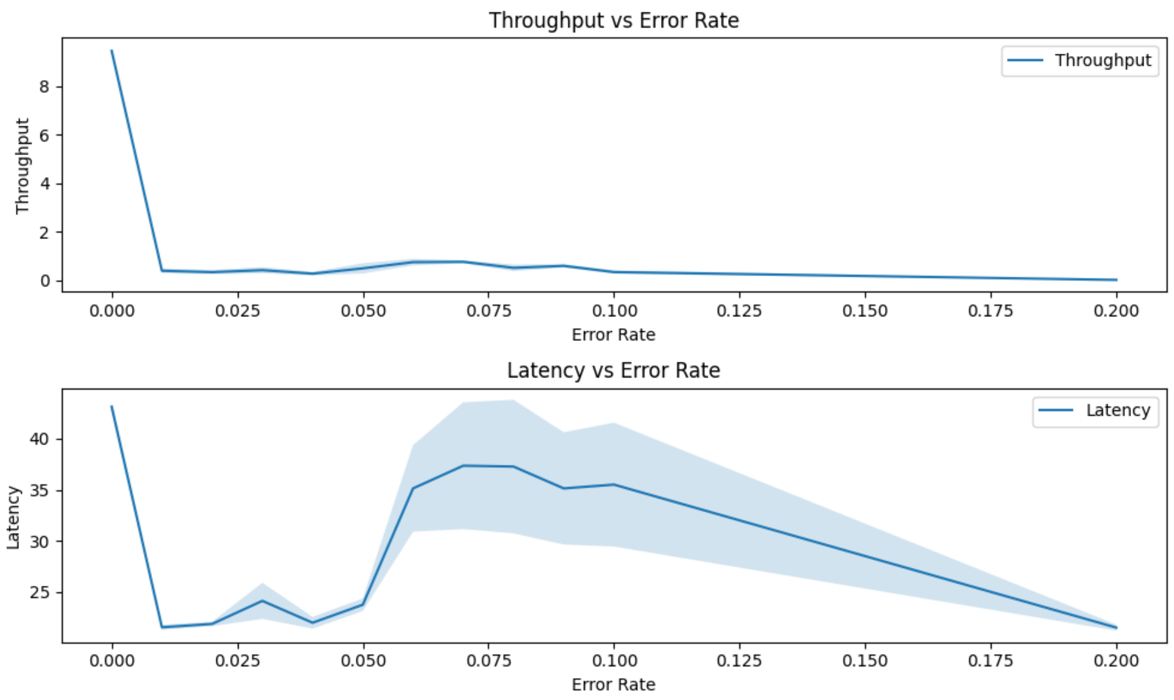
For each batch corresponding to a unique error rate P, we calculated the sample mean and standard deviation for throughput and latency. Using these sample statistics, we constructed 90% confidence intervals for our metrics. The 90% confidence interval for each metric was calculated using the t-distribution since our sample size was relatively small and the underlying population standard deviation was unknown. The t-value corresponding to a 90% confidence level and our sample size (degrees of freedom = number of samples(10) - 1) was used.

The results of these computations are presented in the following table and graph, which show the mean values and their confidence intervals for throughput and latency across different packet error rates. The error rate is varied from 0 to 0.2 to see how it affects throughput and latency. An error rate of 0 means no errors occurred during data transmission, while an error rate of 0.2 implies a higher level of errors.



	Confidence Intervals	
P/ErrorRate	Throughput	Latency
0	(9.382347312768458, 9.513560687231545)	(42.680288185097744, 43.57919181490225)
0.01	(0.2944768974637062, 0.49127950253629393)	(21.281284373494717, 21.77231562650528)
0.02	(0.24803225682492372, 0.4219819431750763)	(21.64402508254065, 22.06973491745935)
0.03	(0.2899947236511129, 0.5443766763488871)	(22.34611981748025, 25.885140182519756)
0.04	(0.1991930737300535, 0.3472175262699466)	(21.403410395832662, 22.54290960416734)
0.05	(0.28026784181201425, 0.7020317581879857)	(23.125296926943953, 24.349143073056048)
0.06	(0.609999594782764, 0.8843570052172361)	(30.897969287426292, 39.370010712573716)
0.07	(0.6990439444739022, 0.820265655526098)	(31.148170241430773, 43.57252975856922)
0.08	(0.3760738404636902, 0.6494351595363098)	(30.731101979287622, 43.815358020712374)
0.09	(0.5247403831152129, 0.6642304168847872)	(29.640455708946526, 40.624604291053465)
0.1	(0.2662937266605278, 0.40964967333947216)	(29.4404093744757, 41.5690306255243)
0.2	(0.0073340992528758, 0.0242986127471242)	(21.20036913751992, 21.79041086248008)

More details are mentioned in simulation_results.csv file which is attached along with all the programs used to simulate and analyze the data.



Above graph is generated using Pythons matplotlib library which compares throughput and latency versus error rate during each batch.

The top graph shows the throughput versus error rate. You can see that as the error rate increases, the throughput generally decreases. The shaded area represents the confidence interval, indicating the possible range for the true average.

The bottom graph shows latency versus error rate. Unlike throughput, there isn’t a clear trend of increase or decrease in latency as the error rate changes. The shaded area here also indicates the range of the confidence interval for the average latency.

5) Conclusion

In conclusion, our simulation study using the ns-3 network simulator has provided valuable insights into the performance characteristics of a TCP/IP network under varying conditions of packet loss rates. Through a series of designed simulation runs, we have been able to quantify how the error rate, acting as the primary factor in our experiments, impacts the network's throughput and latency. We have learned that while TCP/IP networks exhibit resilience to errors through error correction and flow control mechanisms, there is a limit to how much they can compensate for.

In general, we observed that throughput decreases with an increase in packet error rate. This happens because TCP has built-in error control mechanisms that detect lost packets and initiate retransmissions. We saw that the impact on latency isn't always linear or predictable, as it may also depend on factors like the patterns of packet loss, the efficiency of the network's error handling mechanisms and the network topology but latency tends to increase as the error rate goes up due to several factors.