Dr. Matt Mathis's hypothesis
Hypothesis

Dr. Matt Mathis hypothesis: TCP Reno implementations obey the following throughput equation: throughput = (MSS/RTT)*(C/sqrt(p)), where C is a constant, p is the loss probability, RTT is the round-trip time and MSS is 1400 bytes in our project. The following measurements will help you verify this hypothesis.

(References: https://dl.acm.org/doi/10.1145/263932.264023)

Experiment

Use tcconfig to adjust the loss probability p on the sender node to different values.

“sudo tcset eth0 --rate 10Mbps --delay 20ms --loss 0.01%”

For different values of p, transfer a large file and measure the duration of the transfer (make sure to adjust the file size so that your transfer takes at least a few seconds). Use the transfer duration and the file size to calculate the throughput. Repeat the same experiment 10 times for each loss probability. Create a line plot with 1/sqrt(p) on the x axis and throughput on the y axis. Do linear regression and include the regression line on the same plot.

Inference

Based on the results of your experiments, we ask that you answer the following questions:

- Using your linear regression, what value did you find for C?
- Compute the Pearson Correlation Coefficient between 1/sqrt(p) and the throughput that you measured.
- Does your data corroborate Dr. Mathis's hypothesis? Why or why not?

Hint

Due to the lack of the timeout retransmission, sometimes your program may stuck please try again.

Your own CCA

You now have the opportunity to design your own CCA with the goal of outperforming TCP Reno and you can modify anything you like.

- The metric is throughput (as measured by the time it takes to transfer a 1MB file).

Hint

- Test link: RTT = 200ms, the buffer size of the router is infinite. But the router will add extra delay Xms when the buffer depth exceeds N.
  For example, if N = 5 and there are currently 10 packets in the buffer, extra delay will be added for packets 6 through 10.

How to test by yourself

tcconfig:

You can simulate the link (bandwidth, latency and loss rate) by tcconfig.

“sudo tcset eth0 --rate 10Mbps --delay 20ms --loss 0.01%”

burst:

You can simulate the burst traffic by starting an another file transfer between two VMs.

duplicate ACKs:

You can manually send duplicate ACKs at your receiver to simulate the congestion scenarios.

Report

The report should include two parts: (1) Dr. Matt Mathis's hypothesis and (2) Your own CCA.

Dr. Matt Mathis's hypothesis

- Create a line plot with 1/sqrt(p) on the x axis and throughput on the y axis.
- Do linear regression and include the regression line on the same plot.
- Using your linear regression, what value did you find for C?
- Compute the Pearson Correlation Coefficient between 1/sqrt(p) and the throughput that you measured.
- Does your data corroborate Dr. Mathis's hypothesis? Why or why not?

Your own CCA

In your report include a section called Algorithm Proposal. In this section, you should:

- Propose a new algorithm (or a modification to Reno) that improves its throughput. Provide a detailed algorithm description, including why your new algorithm will improve throughput relative to your Reno implementation.

In your report include a section called Algorithm Evaluation. In this section, you should:

- Provide data comparing your Reno implementation to your new algorithm completing a 1MB transfer.
- That means how long does it take for a single connection to transfer 1MB.

Testcase:
1.1) test_noloss (test_file_transfer.TestCases)
Testing: reliable transfer of 50KB file without loss

2.1) test_recv_window_change (test _flow_control.TestCases)
Flow control test - this test checks if sender correctly responds to changes in the receiver's advertised window.

3.1) test_retransmission_after_three_dupacks (test_dupack.TestCases)
DupAck - this test checks if packet is retransmitted after 3 dup acks.

4.1) test_slow_start
(test_congestion_control.TestCases)
Congestion control test - this test checks if the congestion window increases exponentially during slow start

4.2) test_loss_response (test _congestion _control.TestCases)
Congestion control test - this test checks if the congestion window decreases in response to loss.

5.1) test_source_structure (test_submission_files.TestCases)