Taylor Daw

3402948

Part 2 test 1 questions:

For the part one test of 10 pps with a 50 byte payload for 10 seconds, exactly 100 udp packets were sent during the transmission. There were additional packets for DNS that I am going to ignore. Each packet sent across the wire contained 92 bytes (50 data, 42 header). For bytes/second 100 * 92 = 9200/10 comes out to 920 bytes per second. 420/920 means that the overhead of the packet is 45%.

Test 2 questions

1000 packets per second, 50 byte payload, 5 second duration

My program falls far short of the expected 5000 packets, it only reaches 3521. The packet creation stays around even intervals of .0011 seconds between packets on average, but there are some outliers that go far beyond that due to jitter in the network. The average data rate is (3521 * 92) / 5 = 64786 bytes/second

Test 3 questions

1 packet per second, 4000 byte payload, 1 second duration

This packet size goes beyond the MTU of the data link layer, and has to be broken down. The data is separated into 1480 bit chunks and then sent across the network, and then at the end the actual packet is assembled in the final frame of the transmission. (X/1480) * header size + (x%1480) + header size

Test 4 question

When you attempt to send a 100000 byte packet, an error is thrown from socket saying that the packet is too large to be stored in the internal buffer.

Part 3

To look at the loss of each simulation, I decided to filter out everything from the capture and then by using the wireshark statistics summary I can see how many of the packets were received.

To measure jitter, I ran a tshark call **tshark –T fields –e frame.time_delta_displayed –r <file_name>**, and this shows the time between each packet to measure the jitter.

I tested various packet sizes to get a good gauge of where the size started to affect the stream. Running at a constant 1000 pps, it took all the way up to a 3000 size packet before I started to see an effect, anything below that all 500 packets were normally received.

Router A, 4000 bytes @ 1000pps

Number of flows	packets received
1	499
2	500
3	498
4	393
5	499
6	500

Router B 4000 bytes @ 1000bps

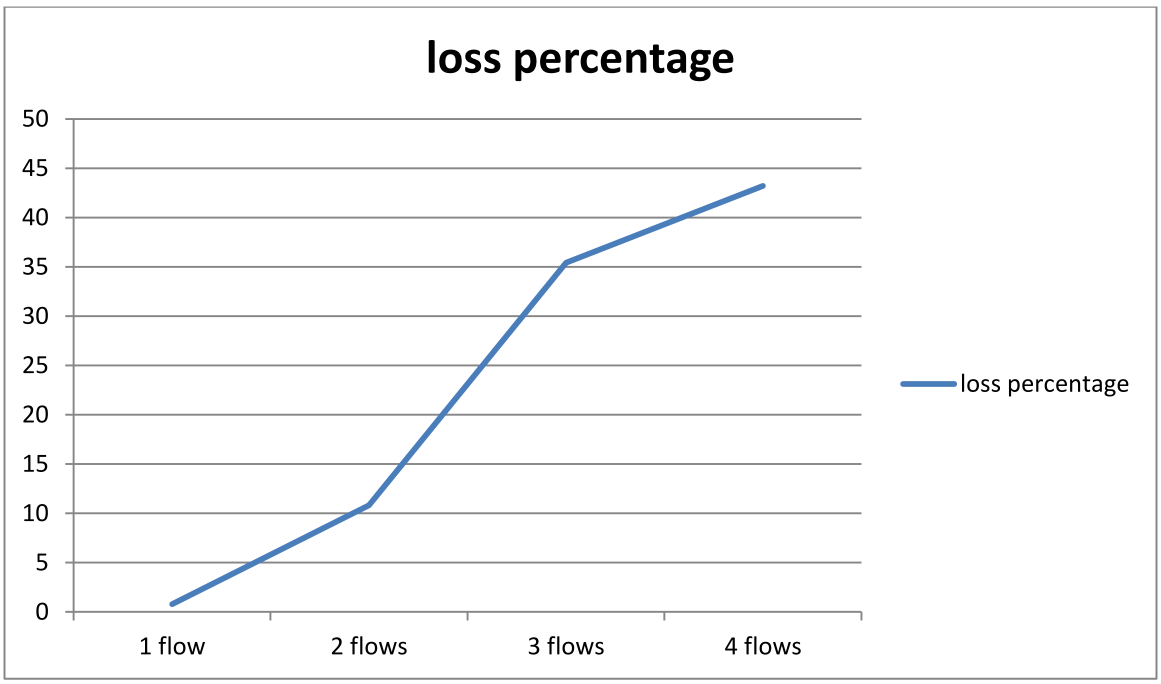
Number of flows	packets received
1	341
2	410
3	404
4	466
5	499
6	487

Unfortunately these results are close to useless. I was doing the testing when everyone else was testing and the traffic was all over the place, so instead of results consistent with what I would expect, they are pretty far off. The one thing that is clear from this is that A is clearly performing better then B, the video quality of B is much lower than A. The number of flows did not have the effect that I expected to see, but once again this is due to other traffic on the network. In the case of B, it should have been exactly the opposite.

A second packet size proved to be more useful

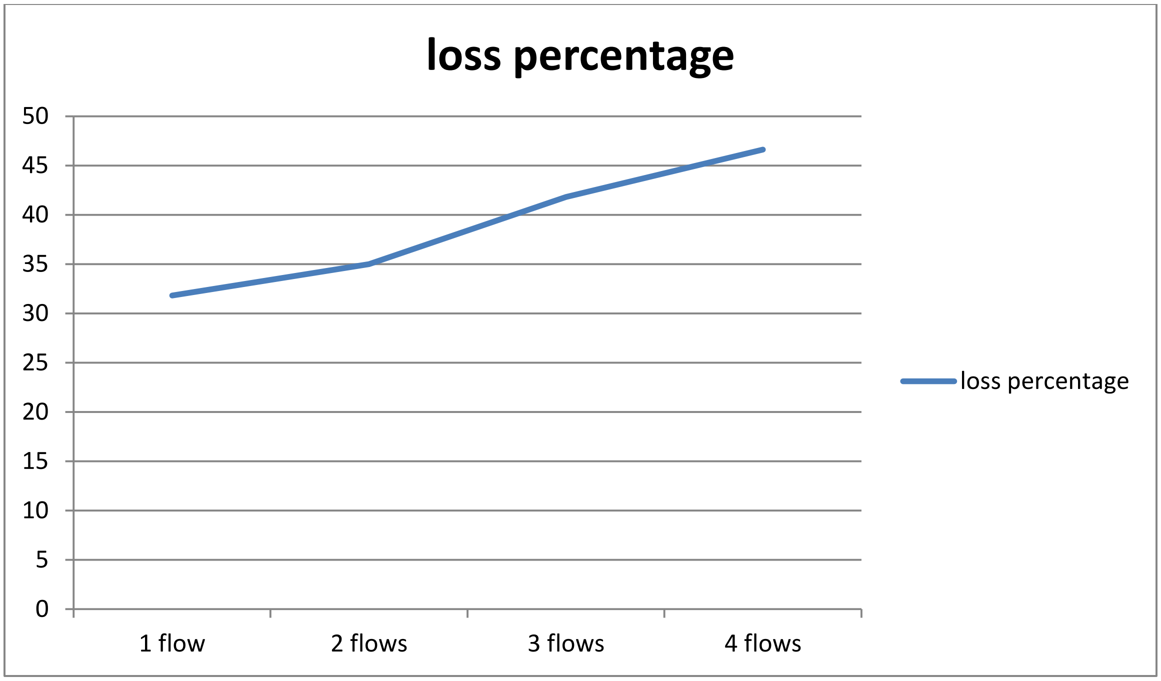
Router A 6000 byte payload @ 1000 pps

Number of flows	packets received	loss percentage
1	496	.8
2	446	10.8
3	323	35.4
4	284	43.2



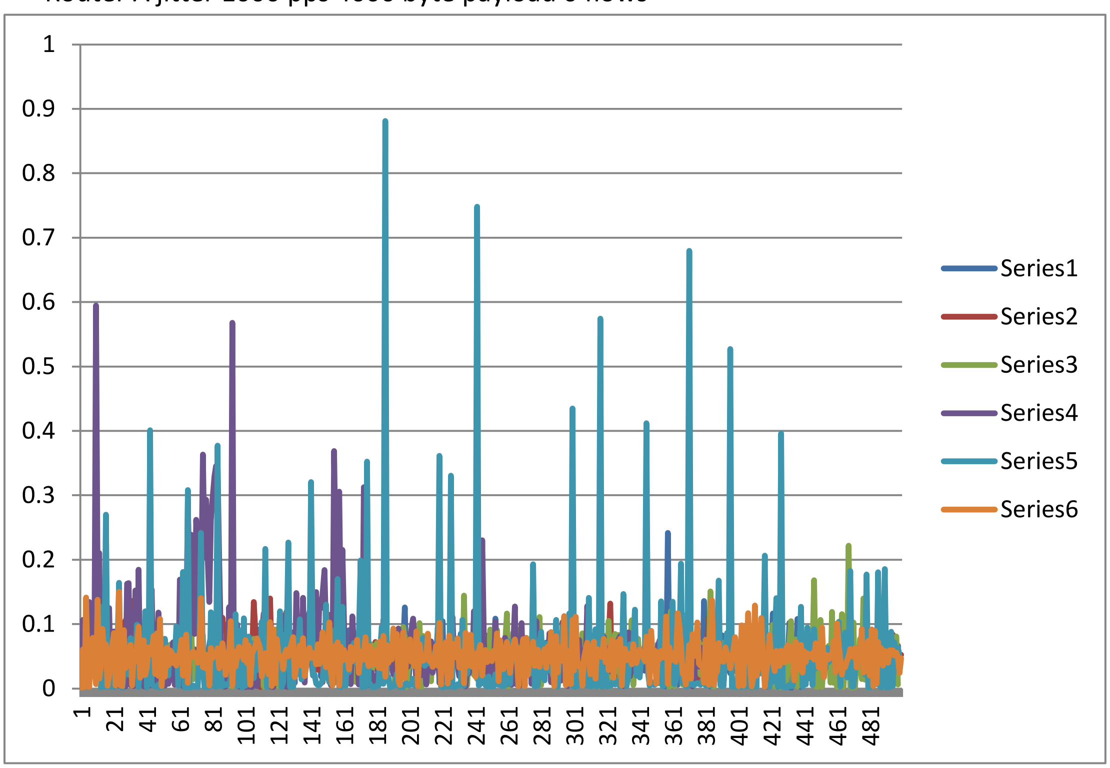
Router B 6000 byte payload @ 1000 pps

Number of flows	packets received	loss percentage
1	341	31.8
2	325	35
3	291	41.8
4	267	46.6



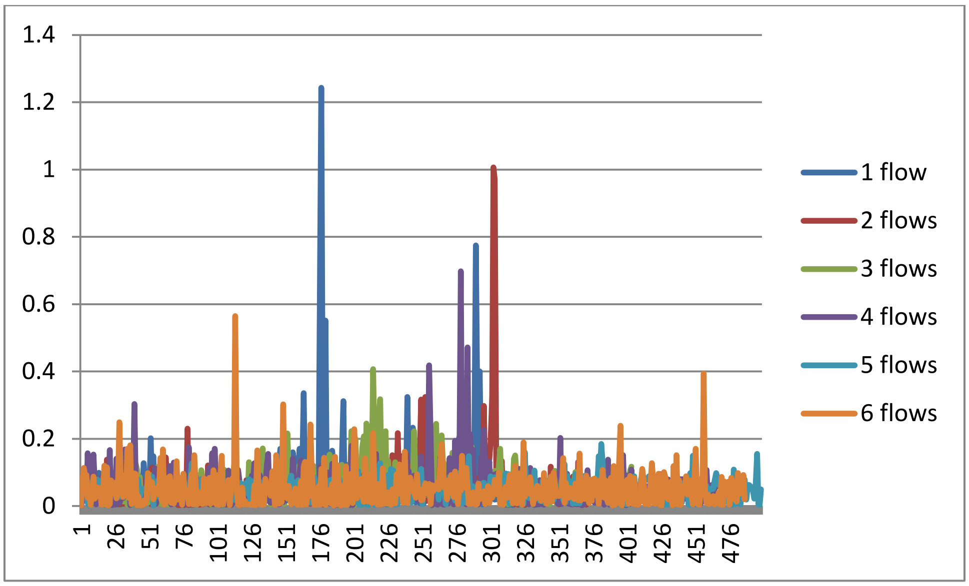
These results show that the number of flows clearly affects the results and impacts the quality of the movie. The consistent results of the B router receiving less packets would suggest that it is not using QoS. Router A is using QoS so that it can prioritize and keep the stream alive better.

Router A jitter 1000 pps 4000 byte payload 6 flows



This graph shows the 6 flows and the jitter between each packet sent. The real delays happen when there are 5 flows going, even though the most should be happening when there are 6. The data is somewhat consistent with the exception of 6, we see that when there are 4 flows there is also some delay as well. The jitter appears to stay relatively low at all times though, typically staying below .1 seconds.

Router B jitter 1000 pps 4000 byte payload 6 flows



This graph seems to have a lot of random delay with almost all cases right in the middle of the transmission, but all tests seem to have relatively the same average jitter during transmission.

With multiple RTP flows on router A it begins to crash rather quickly with any significant traffic. It doubles the required information and thus has problems handling the requests.

In conclusion, A uses QoS and delivers better all around quality under stress from network traffic. When the number of flows increases there is a definite increase in dropped packets, and the delay between packets picks up too. At about 3000 bytes per packet it starts to affect the network, slightly decreasing the quality of the movie. The results are not exactly as expected, but the second round of testing with a larger packet were much more successful and are how they should be.