Simulation based performance analysis

Mayur Joshi **Sarang Hattikar**

joshi.may@husky.neu.edu

hattikar.s@husky.neu.edu

College of computer and information science

Abstract:

Primary objective of this paper is to measure relative performance of four different TCP variants. These include Tahoe, Reno, NewReno, Vegas. Experiments were performed simulating TCP variants using ns-2 (Network Simulator). Analyses of the results obtained from these experiments are presented in this paper.

1. Introduction:

TCP is an important protocol in network protocol stack. It is a connection oriented protocol and hence provides connection oriented reliable services. TCP is one of the most widely used protocol. Along with reliable data transfer, TCP provides various other services such as flow control, congestion avoidance.

Variation in implementation of these services produces variants of TCP. These TCP variants perform differently in various network states. A comparative analysis of these variants helps to determine factors that affect their performance. Simulation based analysis helps in comparative analysis.

In this paper, we will be analyzing performance of four common TCP variants these are Tahoe, Reno NewReno and Vegas. The basic TCP algorithm along with improvement such as slow start, congestion avoidance and fast retransmit is the TCP Tahoe. TCP Reno along with fast retransmit also implements fast recovery. New Reno eliminates Reno's wait for retransmission of packet when multiple packets are lost. Vegas implements delay based congestion avoidance.

Three different experiment are performed first of which measures performance (in terms of throughput packet drop rate and latency) of each of these variants individually in a congested network, second experiment analyzes the bandwidth distribution when two of these variants are implemented simultaneously in the same network. Third experiment analyzes the effect of queuing algorithm such as DropTail and Reno on performance of two TCP variants, Reno and SACK.

2. Methodology:

We used ns2 i.e. network simulator2. Ns2 is a discrete event based network simulator which is developed using C++. It uses object- oriented dialect of tcl. NS2 logs different events of simulation in the into a trace file. This trace file is then parsed using AWK script to determine performance variable such as throughput and latency. Following diagram shows the network topology used in all the three experiment

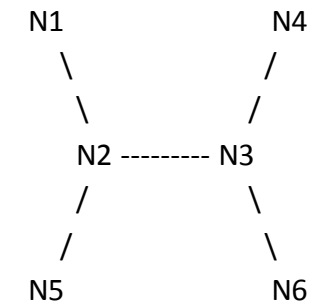

Figure 3(a): Our network topology

As shown Network contains six nodes each and five links. Ech link is TCP duplex 10Mbps link. Following is description of Topology:

1. N1 is TCP source with an ftp applicaton.

2. N2 is CBR (Constant Bit Rate) generator node. Unresponsive UDP flow.

5. N5 is a TCP source.

6. N6 is a sink for TCP source at N5.

For first experiment TCP source at N1 and TCP sink N4 is used. First of all a CBR flow is created followed by a TCP flow. Bandwidth is then incremented in a stepwise manner from 1Mbps to 10Mbps. Performance of TCP flow is then analyzed under such congestion conditions.

For Second Experiment along with topology used for first experiment an additional TCP source and sink is created at node N2 and N3. A CBR flow is first created followed by two TCP flows. Performance of these two TCP flows is measured in presence of other. Four runs of the second experiment are carried out with four pairs of TCP variants such as Reno-Reno, Vegas-Vegas, NewReno-Vegas, NewReno Reno. During each variation CBR flow rate is incremented from 1Mbps to 10Mbps.

For third experiment only one TCP flow is used from N1 to N4. CBR fow rate is also kept constant at 7Mbps. In four runs of the experiment two different TCP variant Reno and Sack are implemented with each of queuing algorithm DropTail and RED

3 Result Analysis

3.1 Experiment 1: TCP Performance Under Congestion

This experiment was aimed at analyzing performance of different TCP variants under congestion. The performance was analyzed on the basis of following parameters i) Throughput ii) Packet drop rate iii) Latency. Detail statistical analysis of outcomes of this experiment can be used to determine the overall best TCP variant and stack up the TCP variants on the basis of their performance. Following is graph of throughput of various TCP variants against CBR(Constant Bit Rate) which is implemented in experiment as a pure UDP stream.

3. N3 is CBR sink.

4. N4 is sink for TCP source at N1.

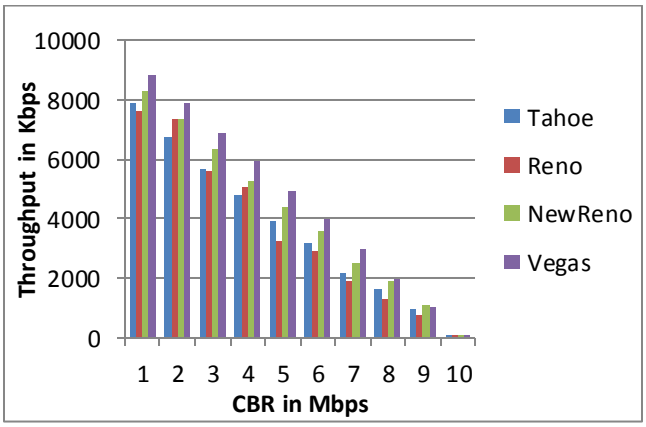

Fig. Shows Throughput of TCP variants versus CBR flow rate

From the above plot it is can be seen that TCP variant Vegas shows the best performance among all the TCP variants. Throughput for Vegas is always higher than all the other TCP variants irrespective of CBR flow rate. Next best TCP variant is NewReno. For initial slow CBR rate throughput of Reno and NewReno is approximately similar while as CBR increases New Reno dominates over Reno. At higher cbr rate on other hand Vegas and NewReno deliver consistent performance. At higher CBR rate packets drops increases. Thus when multiple packets are lost from a single window NewReno can recover without a retransmission timeout.

From above the plot it can be also be verified that TCP variant Vegas, NewReno shows a linear decrease in the throughput as CBR flow rate increases. While Reno does not show linear decrease. This nonlinear decrease in throughput of Reno can be explained by Fast Retransmit Algorithm which implements a varying size of congestion window in its optimized version. NewReno eliminates Reno's wait for a retransmit timer when multiple

Vegas also show the best performance in case of latency.

.

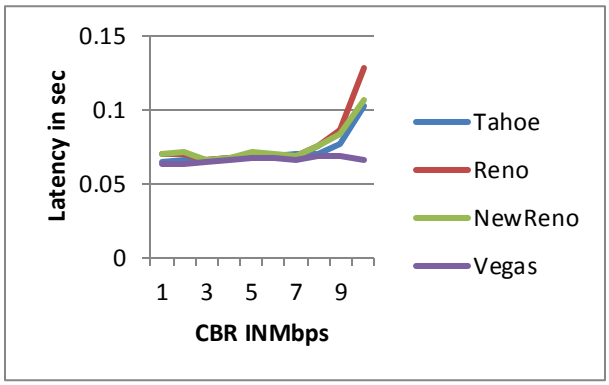

Fig. Shows Latency of TCP variants versus CBR flow rate.

From the above graph it can be concluded that Latency for TCP variant Vegas is minimum among all the TCP variants. For most of the CBR flow rate latency value of Vegas is least among all

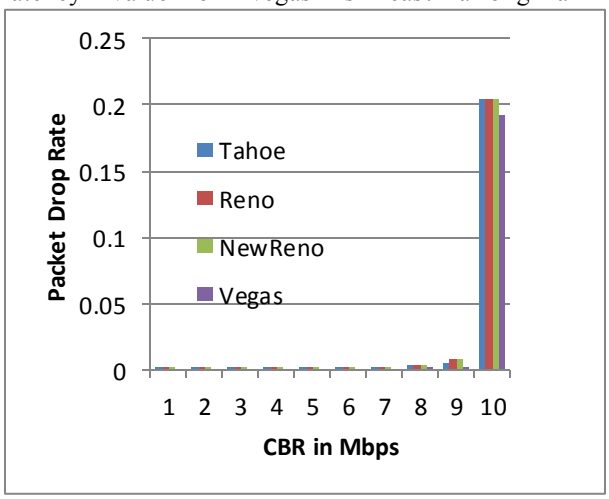

Fig. Shows Packet Drop Rate of TCP variants versus CBR flow rate

Above graph displays packet drop rate of different TCP variants. Here also packet drop rate of Vegas is least among all the TCP variants for the complete range of CBR flows. For very small rate of CBR flow packet drop rate of all the TCP variants is very low and negligible. But as CBR flow rate increases and reaches to bottleneck packet drop rate of all the TCP variants also increases. As we can see just before the bottleneck conditions (at 9mbps cbr rate) Tahoe Reno and New Reno shows a significant increase in packet drops, Vegas shows very little increase in number of packet drops. Even at bottleneck conditions Packet drops of Vegas are lest among all the TCP variants.

Thus among all the TCP variants considered above, Vegas shows the best performance for all the parameters.

3.2 Fairness between TCP Variants

In this experiment two TCP variants are implemented on the same network to determine performance of one TCP variant under the presence of other. Objective of this experiment is to check weather two TCP variant implemented in the same network shares network resources fairly.

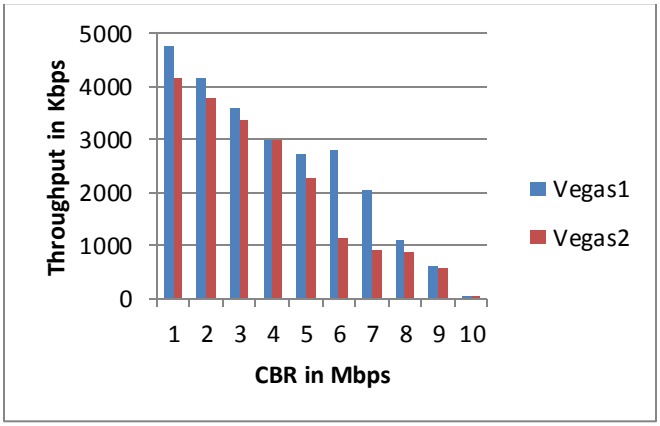

Fig. Shows throughput of Vegas TCP variant along with other TCP Vegas variant versus CBR rate.

From above graph it can be observed that throughput of Vegas1 which has source node as N1 dominates over throughput of other Vegas2 with extremely small margine. Although for most of the cases difference between throughputs is negligible while in a very few cases throughput of vegas1 shoots up. Thus Vegas TCP variant is fair with Vegas.

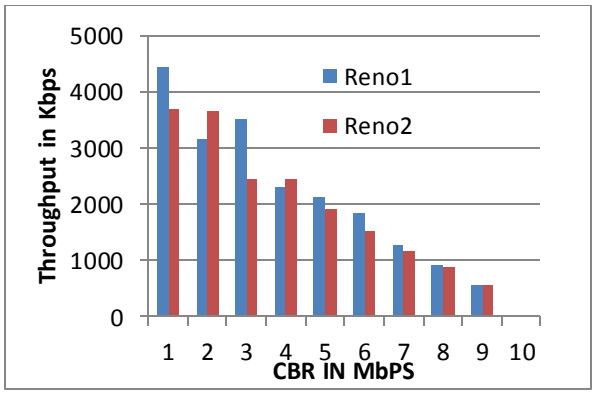

Fig. Shows throughput of TCP variant Reno along with other TCP variant Reno versus cbr rate

From the above graph it is seen that there is no clear dominance of one TCP Reno over other. It can be said that average throughput for both of them is same.

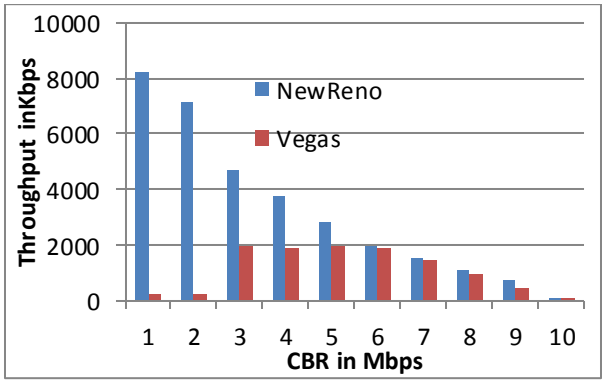

Fig. Shows throughput of TCP variant NewReno along with other TCP variant Vegas versus cbr rate.

From the above graph it can be concluded that NewReno and Vegas do not go well together. In such case NewReno utilizes maximum available bandwidth compared to Vegas. Vegas implements RTT for computation of network congestion. So on sensing congestion due to another TCP source and CBR it reduces its throughput.

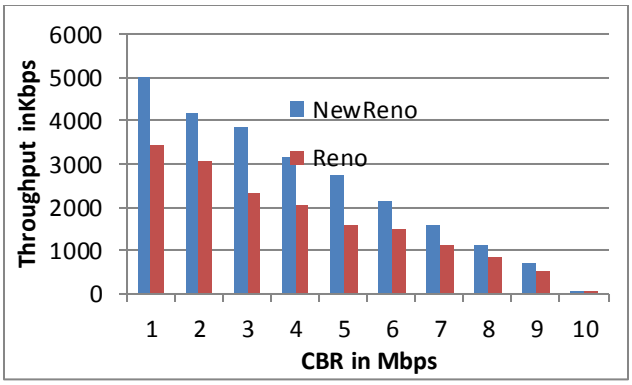

Fig. Shows throughput of TCP variant NewReno along with other TCP variant Reno versus cbr rate

As it can be seen from above graph when When two TCP variants are NewReno and Reno. NewReno always dominates over Reno. NewReno uses maximum of available bandwidth. This can be explained with different congestion algorithm NewReno and Reno implements.

When packet loss occurs Reno waits for 3 consective duplicate ACK. which is the main reason for decreasing throughput of Reno. As NewReno eliminates this wait NewReno's throughput increases.

3.3 Influence of queuing

Performance of a TCP variant is also a function of which type of queue is implemented at the nodes in the network. Aim of this experiment is to determine influence of different queuing algorithm on the performance of TCP variant. Here we will consider queuing algorithm such as DropTail and RED on TCP variant Reno and SACK.

For mapping performance of different TCP variant while queuetype is RED first a constant TCP flow of type either Reno or SACK of 10Mbps is setup and after 40 seconds a CBR flow of 7Mbps is setup. CBR flow is then stopped after 60 seconds.

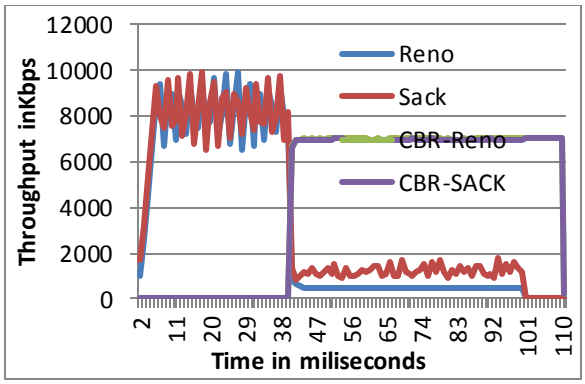

Fig. Throughput of Reno and SACK with DropTail

As we can see from graph during time period when only traffic in the network is of TCP. TCP attains its maximum throughput. As soon as CBR flow starts throughput drops and oscillates between 6000 and 10000. But as SACK implements fast recovery as soon as its throughput drops it again adjust its congestion window thus its throughput again increases.

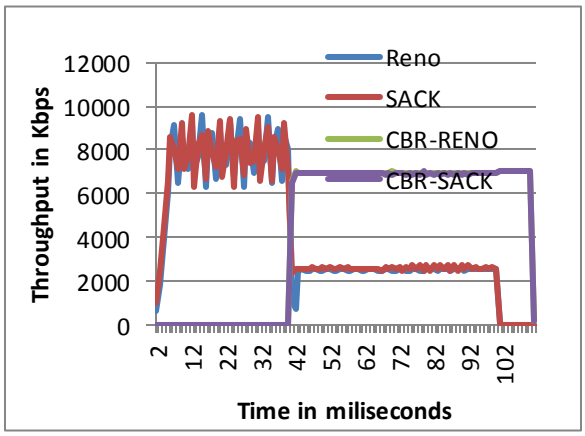

Fig. Throughput of Reno and SACK with RED implementation

As based on throughput certain we cannot decide which queuing algorithm is better. Lets consider other parameters:

	DropTail	RED
Reno	0.00154448	0.00061112
SACK	0.00160811	0.000616975

Table(1) Packet Drop Rate

Latency varies as follows:

	DropTail	RED
Reno	0.0655081	0.065285
SACK	0.0658671	0.0651426

Table(2) Latency Variation

As it can be seen from result that droprate and latency is least in RED implementation with both of the TCP variants.

Throughput of SACK with RED implementation is constant and greater than DropTail Implementation. Latency of SACK along with RED is least among all the variation. Thus in can be concluded that RED implementation forms a good TCP variant.

Conclusion:

Objective of the first experiment was to analyze performance of various TCP variants. So from first experiment we can conclude that VEGAS is the best TCP variant among all TCP variant considered above. From second experiment it can be concluded that TCP variants are relatively fair to themselves. But a TCP variant does not provide fair bandwidth distribution with other TCP variants. Worst pairing among all was Vegas and NewReno where NewReno utilizes most of the available bandwidth. NewReno and Reno pairing was relatively better. Still in this pairing also NewReno dominated Reno in bandwidth utilization. Last experiment was aimed at measuring influence of queening algorithm RED and DropTail on performance of different TCP variants. While average throughput in both the cases was almost same. End to end latency in case of RED is lower than DropTail.

4 References

1. “Simulation-based Comparisons of Tahoe, Reno, and SACK TCP” by Kevin Fall and Sally Floyd

.