Simulation of Snoop protocol for TCP

Group - 24

Submitted By:
K Vijaya Kumar(09305081)
Neelamadhav Gantayat(09305045)
Raju Chinthala(09305056)
Kusuma Haranath P(09305046)

November 13, 2009

Problem Statement :

Simulation of Snoop protocol for TCP using NS2. The Transport Control Protocol (TCP) is a reliable protocol designed to perform well in networks with low bit-error rates, such as wired networks. TCP assumes that all errors are due to network congestion, rather than to loss. When congestion is encountered, TCP adjusts its window size and retransmits the lost packets. In wireless networks, however, packet loss is mainly caused by high bit-error rates. Thus, the TCP window adjustment and retransmission mechanisms result in poor end-to-end performance.

Several solutions have been proposed to the above stated problem like Freeze TCP, Mobile TCP and Indirect TCP. We will focus our study on Snoop Protocol or what is at times called as TCP-aware-link layer protocol. First we will describe the basics of the Snoop protocol. Then we will present the experimental results of our simulations of this protocol. All simulations were performed using Network Simulator 2 (NS2). These results will show how the use of Snoop protocol can greatly enhance the performance of TCP over single hop wireless networks using wired infrastructure. Finally we make our concluding remarks.

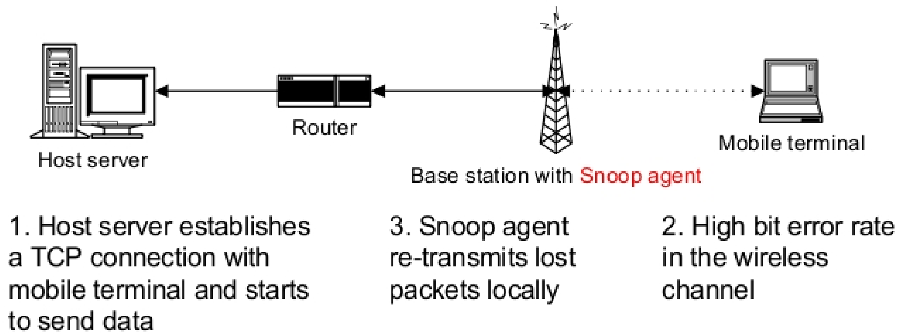
Snoop Protocol :

This protocol was proposed by Hari Balakrishnan, Srinivasan Seshan,Elan Amir and Randy H. Katz. The Snoop protocol runs on a Snoop agent that is implemented in a base station or a wireless device. The agent monitors packets that pass through the base station and caches them in a table. After caching, the agent forwards packets to their packets destinations and monitors the corresponding acknowledgements.

The role of the Snoop agent is to cache data packets for a TCP connection. When data packets are lost (indicated by the reception of duplicate Acks), the Snoop agent retransmits those packets. It retransmits them locally without forwarding the Duplicate ACKs to the sender. Hence, since the TCP layer is not aware of the packet loss, the congestion control algorithm is not triggered. In addition, the Snoop agent starts a retransmission timer for each TCP connection. When the retransmission timer expires, the agent retransmits the packets that have not been acknowledged yet. This timer is called a persist timer because, unlike TCP retransmission timer, it has a fixed value. Figure 1 illustrates a Snoop agent implemented in a base station.

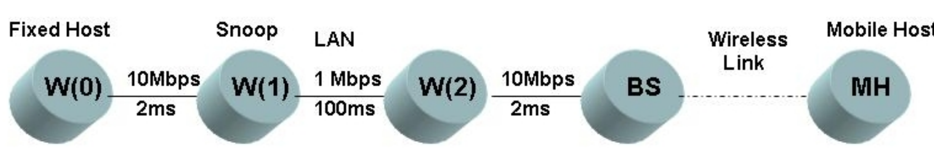
Experimental Scenario :

Since snoop support is available only for LAN not for wireless network in NS2 [1] so we have designed a scenario(shown in figure2) to implement it on wired cum wireless network. We have created hieririchical routing to support wired and wireless network. In the network there are two domains and in each domain one cluster is created. in the first domain and first cluster wired network with previous texts)



lan is created. and in the second domain the wireless node with the base station is established. In this scenario we are creating a two node LAN with high bandwidth and very low delay, we are implimenting snoop agent at node1 of LAN and node2 is connected to Base Station through a wired link of high bandwidth and very low delay.

A mobile node is connected with the base station through lossy wireless link, to implement this link we are attaching a error module with the wireless link. Here the incoming error procedure is included. The incoming error module lets each receiver get the packet corrupted with different degree of error since the error is independently computed in each error module. Sender node is connected to node1 through a wired link having 5MB bandwidth and 100ms delay. Snoop agent is implemented on node1 with



DropTail Queue of length 100. A TCP connection is made between source fix host and destination mobile host, we are using TCP Reno to implement it. A FTP application is running on top of the tcp connection which starts at time 1.0s and stops at 100.0s. We are using very high bandwidth and very low delay for LAN and wired link between node2 and BS so that these links dont give any effect on the traffic between the source and destination. Hence we can assume that the snoop agent and the Base Station both are at same node, which is the case described in figure1.

Here two scenerios were created one with snoop installed in the middle node and the other with out the snoop.

Settings for the both scenarios are same.

Network Settings

Following are the settings of the nodes:

IP addresses

Node W(0) 0.0.0
Node W(1) 0.0.1
Node W(2) 0.0.2
LAN 0.0.3
Node BS 1.0.0
Node MH 1.0.1

Mobile Node Settings:

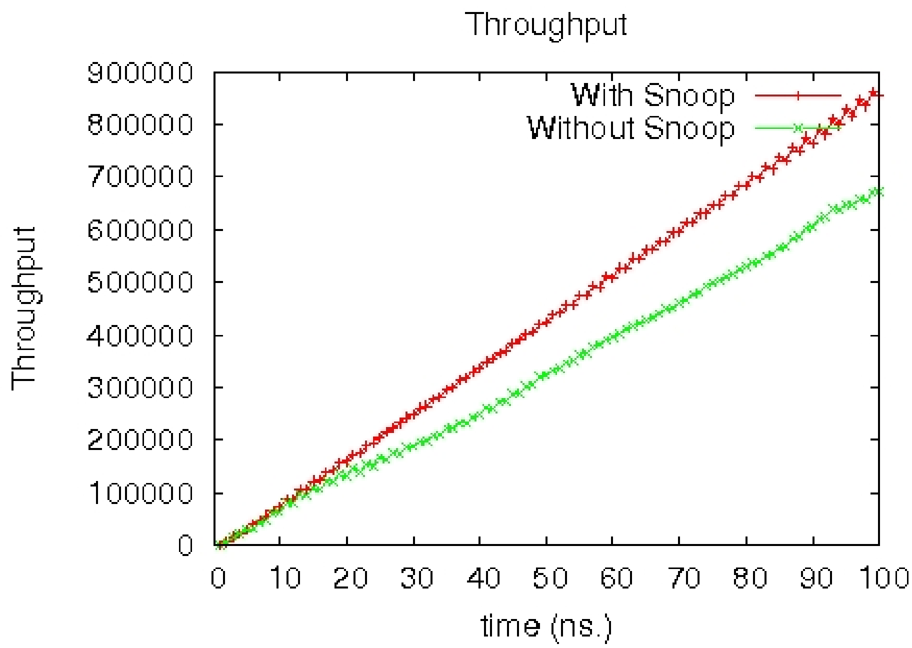
set opt(chan) Channel/WirelessChannel ; channel type set opt(chan1) Channel set opt(prop) Propagation/TwoRayGround ; radio-propagation model set opt(netif) Phy/WirelessPhy ; network interface type set opt(mac) Mac/802_11 ; MAC type set opt(ifq) Queue/DropTail ; interface queue type set opt(ll) LL ; link layer type set opt(ant) Antenna/OmniAntenna ; antenna model set opt(x) 670 set opt(y) 670 set opt(ifqlen) 100 ; max packet in ifq set opt(nn) 3 ; number of mobilenodes set opt(adhocRouting) DSDV ; routing protocol set opt(lls) LL set opt(cp) "" ; connection pattern file set opt(sc) "scen-3-test" ; node movement file. set opt(x) 670 ; x coordinate of topology set opt(y) 670 ; y coordinate of topology set opt(seed) 0.0 ; seed for random number gen. set opt(stop) 250 ; time to stop simulation set opt(bw) 11Mb ; Bandwidth set opt(delay) 2ms ; Delay set opt(stop) 5 ; Simulator stop time

Experimental Results

Here The results were extracted from trace files generated by NS2 for the scenario described above. Perl scripts were used for the purpose. The following results will demonstrate that the use of snoop greatly improves the performance of TCP over wireless link.

Change in Throughput Over Time

The following two graph illustrated below show the change in Throughput of TCP Reno with and without snoop over time. For low error rates, regular TCP performs much the same as snoop pro- tocol. Whereas, for high error rates, performance of regular TCP drops drastically. Snoop protocol performance in case of high error rates is better than regular TCP as shown by the graph.

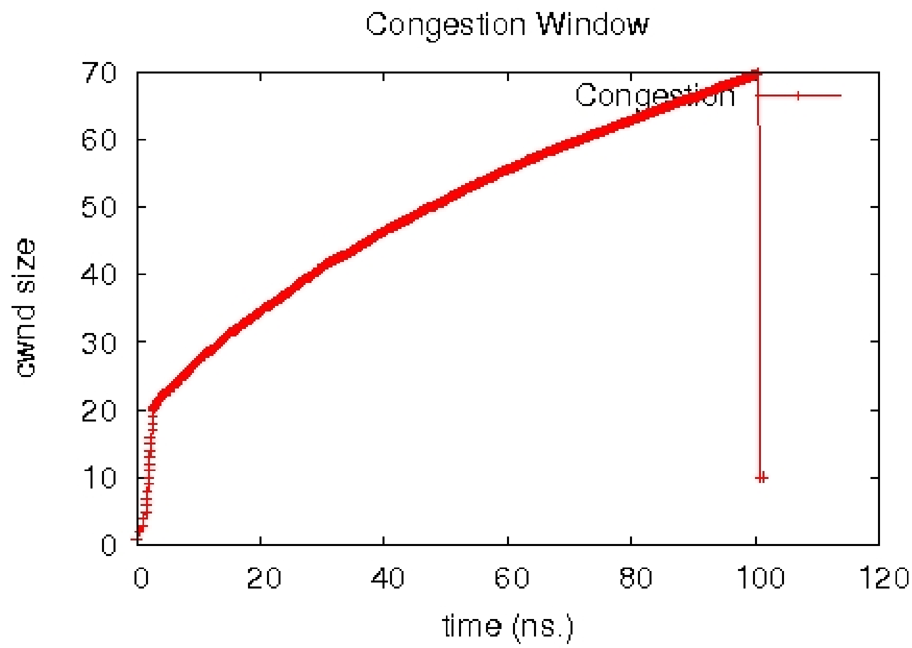


Change in Congestion Window over time

How the congestion window size changes with time for the Regular TCP protocol and the Snoop TCP protocol can be seen in figure 6 and figure 7 respectively. It is obvious that Regular TCP is not able to grow its window size due to the link errors and this explains its low throughput. Snoop TCP has a higher congestion window size on an average which makes it perform better.

Comparison using Sequence Traces

TThe figure 8 shows a comparison of sequence number progression with time in case of regular TCP and snoop protocol. The error rate in both cases is kept at 18%. The graph clearly shows that snoop protocol does better than regular TCP. In regular TCP, since dupacks for missing packets are forwarded to the TCP sender, the sender initiates slow start by reducing the congestion window size, increasing retransmission timer duration, because of which, in a given time interval, due to more retransmissions, less new data is sent. Hence, growth of sequence number is lesser than that in case of snoop protocol. In snoop protocol,



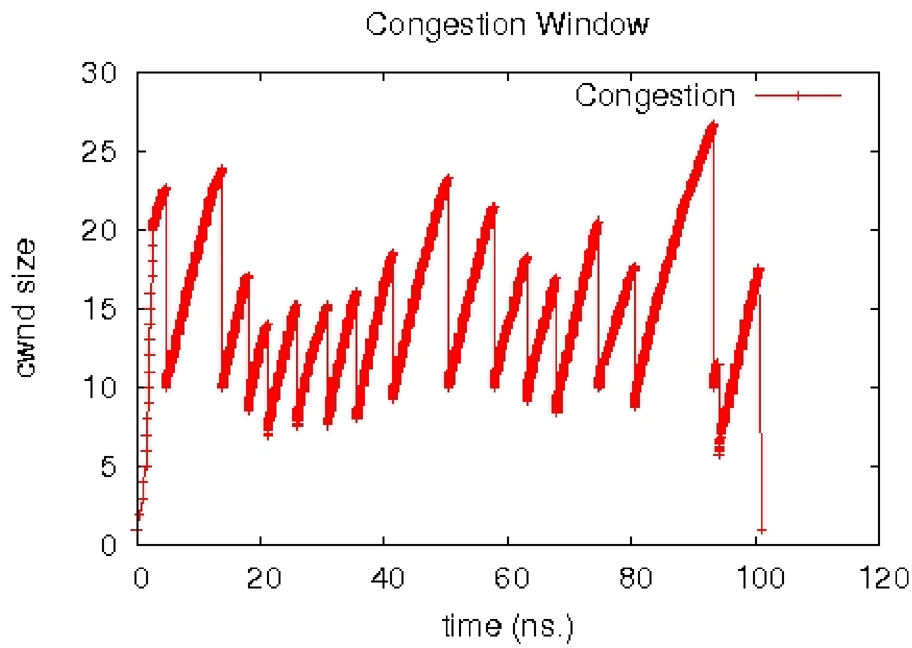
the retransmissions are carried out by the snoop agent. Since, the snoop agent does not forward the dupacks back to the sender, sender is always sending new data to the MH. Hence, growth of sequence number is more. Also, the growth rates decrease as the error rate increases and this rate of decrease is high in case of regular TCP.

Performance benefits of Snoop

The Snoop protocol as can be clearly seen from figure 9 shows substantial benefits over normal TCP. Snoop TCP shows an interesting characteristic here. That is the performance benefits are significantly higher for moderate wireless error rates. For low and very high error rates the gains are relatively marginal. This is because for low error rates normal TCP does a decent job as is expected and snoop gives it no added benefit. And for high error rates 9 due to increased loss probability of the retransmitted packet by the snoop agent wireless link, there will be cases when the sender times out.

Throughput Calculation:

We have written a Shell script Script to find out the throughput in Kbits per sec for both the scenarios. It displays both number of packets sent and throughput.

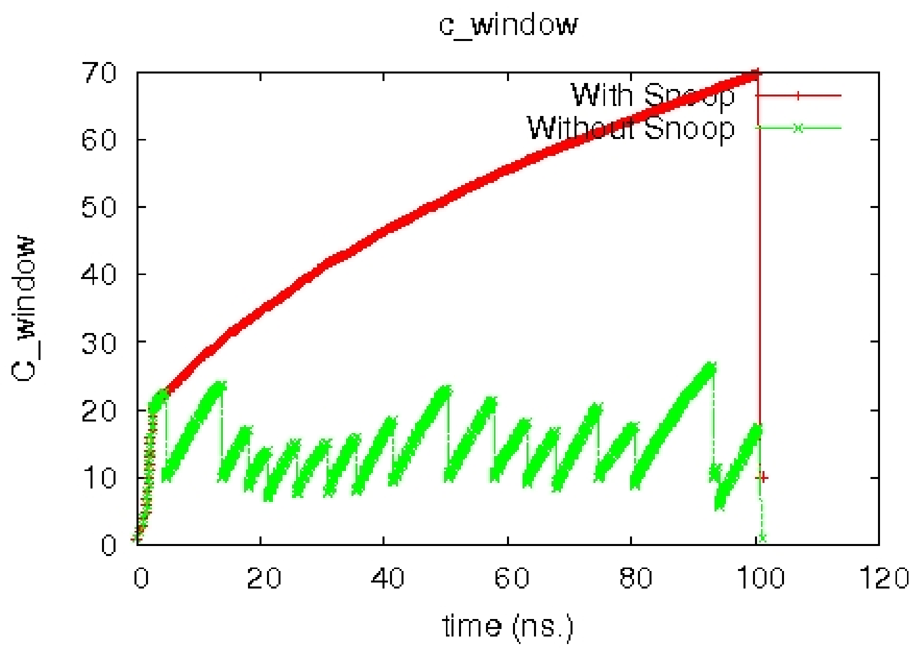


Observations:

- We have found that with snoop TCP and with High Error Rate, number of packets received by the receiver is more during the stipulated simulation time.
- When there is no snoop agent, The throughput of the receiver in the given time is less. This is due to loss of packets and adjustment of congestion window by TCP Reno
- With snoop TCP though the packets were lost snoop agent will retransmit the packets.

Conclusion:

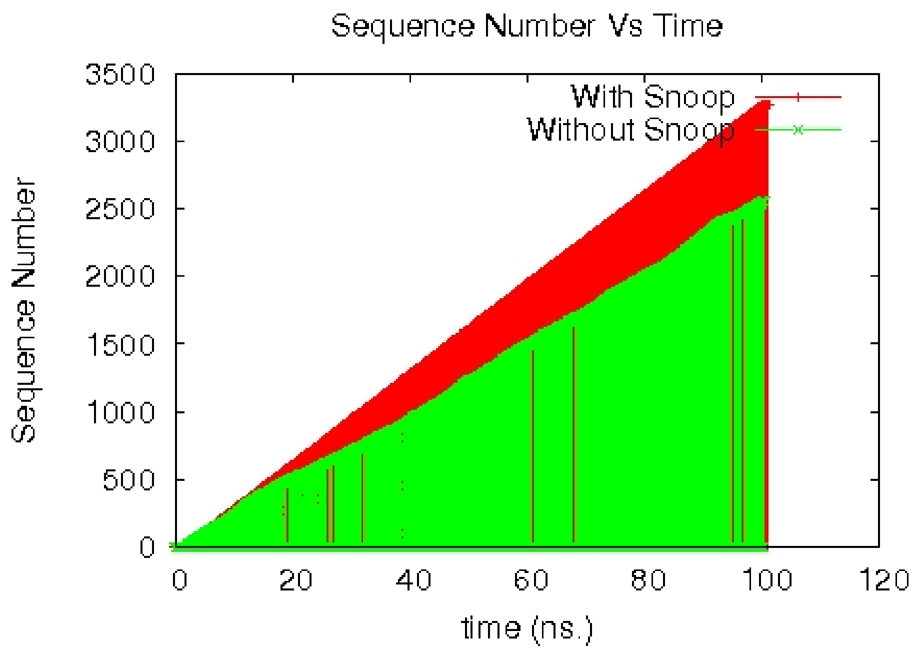
We simulated Snoop protocol using Network Simulator 2 and extracted statistical data from its trace files. We compared TCP Reno with Snoop en- abled TCP using this data. Our analysis was based on various types of graphs that modelled parameters like Throughput, Congestion window and sequence trace for both the protocols. It is clear from from our analysis that snoop protocol



greatly enhances the performance of TCP over wired-cum- wireless network. It has been clearly observed that the snoop agent is able to shield the sender from decreasing its congestion window by misinterpreting wireless error losses as congestion. Also we have been able to conclude that snoop protocol is most efective over moderate error rates.

References:

- [1] Kevin Fall and Kannan Varadhan. The ns manual, 25,November 2007.
- [2] Elan Amir Hari Balakrishnan, Srinivasan Seshan and Rendly H. Katz. Improving tcp/ip performance over wireless networks, 1995.
- [3] George Xylomenos and George C. Polyzos. Tcp performance issue over wireless links, 2001.


Sequence Number Vs Time
With Snoop
Without Snoop
Sequence Number
3500
3000
2500
2000
1500
1000
500
0
0
20
40
60
80
100
120
time (ns.)