A Mathematical Approach to Avoid Congestion and To Analyize Snoop Behaviour In Wired Cum Wireless Network

Maninder Kaur, Parminder Singh

Abstract: Performance of the TCP (Transmission Control Protocol) has been promising in wired networks. In wired network the packet loss is due to congestion. But the performance of TCP has degraded in wireless network where packet loss is not only due to congestion but to be also due to high bit error rates and hand offs. Also improving its performance in wired-cum-wireless networks preserving the end-to-end nature of TCP is a difficult task. To address this issue, several new protocols and TCP modifications have been proposed. Snoop is one such modification. In this paper we have surveyed some of the proposed solutions to improve TCP performance on wired-cum-wireless medium.

Index Terms—Snoop Protocol, TCP, Snoop Module, wired-cum-wireless networks, Congestion.

I. INTRODUCTION

TCP was designed to run over any packet-switching wired networks. TCP provides reliable service without overloading the network. Popularity of TCP leads the network engineers to apply TCP in wireless technology as well. most Internet applications and services require error-free data delivery and proper sequencing, providing reliable transmission of data from source to destination has become an important issue in the Internet. Therefore the TCP has been implemented. The error control mechanisms, implemented at the lower layers of the protocol stack, cannot fully replace end-to-end error control, since the end-to-end functionality cannot be achieved in a hop-by-hop manner. The Transmission Control Protocol (TCP), a transport layer protocol, provides transparent transfer of data between application layer entities and releases them from any concern with the detailed way in which reliable delivery of data is achieved. To provide reliability, TCP detects errors or lost data and triggers retransmission until the data are correctly and completely received. TCP is also responsible for ensuring that the sending rate is appropriate for the capabilities of the receiving host (flow control), as well as avoiding introducing too much data into the network, which could cause buffers in some bottleneck routers to overflow and start dropping packets (congestion control). This is done by regulating the rate at which the sending host transmits data. Due to such benefits as end-to-end error control and rate adaptation, TCP is heavily used throughout the Internet: about 90% of today's Internet traffic is carried by TCP[1].

Manuscript Received on December, 2012.

Maninder Kaur, Research Scholar, Department Of Information Technology, CEC Landran .

Parminder Singh, Assistant Professor, Department Of Information Technology, CEC Landran.

II. PREVIOUS STUDY

Recently, several schemes have been proposed to alleviate the effects of non congestion-related losses on TCP performance over networks that have wireless or similar high-loss links [3], [7], . These schemes choose from a variety of mechanisms, such as local retransmissions,

split-TCP connections, and forward error correction, to improve end-to-end throughput. However, it is unclear to what extent each of

the mechanisms contributes to the improvement in performance. In this paper, we examine and compare the effectiveness of these schemes and their variants, and experimentally analyze the individual mechanisms and the degree of performance improvement due to each.

There are two different approaches to improving TCP performance in such lossy systems. The first approach hides any non congestion-related losses from the TCP sender, and therefore requires no changes to existing sender implementations. The intuition behind this approach is that, since the problem is local, it should be solved locally, and that the transport layer need not be aware of the characteristics of the individual links. Protocols that adopt this approach attempt to make the lossy link appear as a higher quality link with a reduced effective bandwidth. As a result, most of the losses s seen by the TCP sender are caused by congestion. Examples of this approach include wireless links with reliable link-layer protocols such as AIRMAIL [5], split-connection approaches such as Indirect-TCP [3], and TCP- aware link-layer schemes such as the snoop protocol [7].

2.1 Snoop Protocol:

The Snoop Protocol [6]: The snoop protocol introduces a module, called the snoop agent, at the base station. The agent monitors every packet that passes through the TCP connection in both directions, and maintains a cache of TCP segments sent across the link that have not yet been acknowledged by the receiver. A packet loss is detected by the arrival of a small number of duplicate acknowledgments from the receiver or by a local timeout. The snoop agent retransmits the lost packet if it has it cached, and suppresses the duplicate acknowledgments. In our classification of the protocols, the snoop protocol is a link-layer protocol that takes advantage of the knowledge of the higher layer transport protocol (TCP).[8]

The main advantage of this approach is that it suppresses duplicate acknowledgments for TCP segments lost and retransmitted locally, thereby avoiding unnecessary fast retransmissions and congestion control invocations by the sender. The per-connection state maintained by the snoop agent at the base station is soft, and is not essential for correctness. Like other link-layer solutions, the snoop approach could also suffer from not being able to completely shield the sender from wireless losses.[7][3].

Snoop module keeps track of all ACKs sent from the mobile host. When a packet is lost, which is indicated by a duplicate ACK or a local timeout, the Snoop protocol retransmits the lost packet to the mobile host provided that the packet has been cached. In this way, the protocol hides the loss from the sender at the fixed host by not propagating the duplicate ACK. Hence, the TCP sender is prevented to invoke the congestion control mechanism, which is not necessary because the packet is lost due to the error in link and not to congestion. Snoop can work very well in high error rate networks such as wireless ones.[6][3]

The Snoop module contains two procedures named snoop_data() and snoop_ack(). The snoop_data() processes and caches data packets sent to the mobile host. Whereas, snoop_ack() processes the acknowledgments (ACKs) transmitted from the mobile host. Whenever duplicate ACK or local timeout occurs, snoop ack() monitors and processes the acknowledgments (ACKs) sent by the MH and performs various operations depending on the type and number of acknowledgments it receives.

2.2 Problem Formulation

As we know that the performance of TCP is good in wired networks but TCP performance degrades when it comes to wireless networks. From last years the performance of TCP is research topic for many authors. To improve the performance of TCP in wireless networks ,various mechanisms are used. Snoop protocol is one among these proposals that is used to improve its performance in wireless network also. The reason behind its poor performance is errors occur over the wireless link. These errors occurs due to packet losses over the congested network. The reasons behind the packet loss can be collisions ,mobility ,channel errors ,buffer overflow etc. When we applied snoop the at the base station that liesA between the wired and wireless networks. The throughput is expected to improve by using snoop mechanism because snoop controls congestion over the network by hiding packet losses from the sender. Snoop is good approach that is used these days for improving the throughput of TCP in wireless network. Snoop agent is works at base station because it monitors every TCP packet that passes through the base station in either direction. This snoop agent maintains the cache of all TCP segments transmitted by fixed host from wired network to the mobile host. The snoop agent retransmit the packet to destination when it receives duplicate ack from destination. The buffered segments are removed from buffer when these are successfully acknowledged by the destination. The successful acknowledgements are forward to the sender by snoop agent that lies on the Base station. The problem occurs when large queuing delays are experienced at the destination end due to packet arrival rate. The sender performing unnecessary retransmission in order to compensate for dropped packets due to buffer overflow. Premature retransmission are also creating unnecessary congestion due to min Round trip time setup. These problems can be faced by TCP in wireless network even using snoop mechanism in wired cum wireless networks.

III. LITERATURE SURVEY

Author [8] has discussed The Snoop protocol is one proposal for improving TCP throughput in wireless networks. TCP throughput is badly affected over the wireless networks due to packet losses. Snoop is implemented to the base station to improve TCP throughput in wireless networks by hiding packet losses from the sender. Snoop was performing badly compared with regular TCP even when there were no packet losses or errors. The main cause for this is premature retransmissions performed by Snoop. The Snoop protocol is modified to avoid these unnecessary retransmissions by having a higher local retransmission timeout. The results show us that Snoop benefits from this approach which has made a significant performance improvement over regular TCP in multi hop wireless networks. When a higher retransmission timer is applied it helps to snoop to improve its performance by suppressing retransmissions in multi hop wireless network. As we know that wireless network is more complicated and lossy network than wired network. wireless network is higher error rate than point to point link.[8]
Author[9] discussed TCP (Transmission Control Protocol) has been performing well over the traditional wired networks where packet losses occur mostly because of congestion, it cannot react efficiently in wireless networks, which suffer from significant non-congestion-related losses due to reasons such as bit errors and hand offs. Snoop protocol and their combination can be used to improve the performance of TCP in WI-Max network, ECN will help in congestion control and SNOOP will retransmit the packets that are lost from nodes in between, saving nearly half the retransmission time and avoiding the decreasing in transmission speed. If snoop and ECN both are applied simultaneously then TCP performance is increased in WI-Max networks. As ECN was help in congestion control and SNOOP was help retransmit the packets that are lost from nodes in between, saving nearly half the retransmission time and avoiding the decreasing in transmission speed and an optimum transmission performance in a wireless network can be achieved.
Author [1] K. Pentikousis ,H. Badr [2000] discussed that As New classes of hosts such as mobile devices are gaining popularity, while the transmission media become more heterogeneous. Wireless networks exhibit different characteristics than wired ones. Mobile hosts have different needs and limitations than desktop computers. TCP has served well the wired Internet for almost 20 years but is not ready for wired-cum-wireless environments. The Challenges that has to face by TCP is increase day by day. TCP in wireless environments is a not very attractive choice. solutions proposed for wireless LAN s do not perform well in wireless Wans and vice versa. The various challenges analyzed by author are like Limited bandwidth, Random losses, Long round trip Times and User mobility. New proposals like snoop ,link layer solutions, tcp variants ,new transport protocols were used to demonstrate their improved performance not only in terms of throughput but also in terms of power consumption in order to succeed.

Author [10] has described as TCP is the most widely used transport protocol originally designed for wired networks. But many experiments have shown that its performance is poor when used in wireless networks. Also improving its performance in wired-cum-wireless networks preserving the end-to-end nature of TCP is a difficult task. To address this issue, several new protocols and TCP modifications have been proposed. Snoop is one such modification. But it can not be used in isolation but has to be combined with TCP. Whenever Snoop is added to TCP Sack

or TCP Reno, the performance of the TCP Sack or TCP Reno is observed to be deteriorating when compared with the performance of TCP Sack or TCP Reno alone. In the case of TCP SACK, the Snoop protocol is not capable of interpreting SACK blocks and interferes negatively in the functionality of the protocol. In the case of TCP Reno, the Snoop protocol drops the duplicate acknowledgements which affect the TCP Reno receiver's congestion window size and results in poor performance of the protocol.

In order to address this issue, the Snoop protocol is modified such that it sends duplicate acknowledgements to the sender without dropping duplicate acknowledgements. The modified snoop protocol improves the performance of TCP SACK by around 10% compared to the plain TCP Sack protocol and about 5% in an environment where no TCP enhancing mechanism is in place. The performance of TCP RENO is also improved by around 30% compared to the plain TCP Reno protocol and about 20% with no TCP enhancing mechanism is in place.

Author [11] has discussed about that Most of the Internet content is delivered using TCP and with the advances in wireless and mobile networks, improving TCP performance over heterogeneous networks like wired- cum-wireless networks has been an important issue. The errors that occurs on network creates big problems in communication. These errors are responsible for various types of Losses like packed loss ,packet dropping and handoff problems. Due to this performance of network is decreased. To control this problems author implemented an adaptive approach to recover from multiple transmission drops from the same window. Experiment results shows that this approach has increase the performance of TCP without extra load on the network. This approach achieved that by resending only number of packets equal to the dropped packets which have already left the network. Also helped to cut the congestion window even for non-congestion drops. This also will be part of a complete set of algorithms which are combined will form a complete mechanism to govern TCP end-to-end error discriminators reaction towards transmission drops. All these actions will help even error discriminators with low/medium accuracy to improve TCP performance with no harm to the network. .

Author [12] described about the wireless multi-hop networks, a large congestion window increases the probability of contention and packet losses, and TCP performance is degraded severely as a result. So, it is necessary to limit the TCP congestion window size in order keep the probability of contention loss in the system to a minimum. so the author created a simulation environment using ns2 simulator to improve the performance of TCP by using a new proposed scheme for the optimization of maximum window size based on the measured bandwidth and Round Trip Time(RTT).A new maximum congestion window setting) algorithm is developed for measuring the bandwidth of routing path in multihop wireless network and also its RTT. By using a new scheme to compare the TCP throughput with TCP New Reno and CWL schemes and proposed scheme performs better under various conditions like traffic condition and different topology networks.[12].

IV. OBSERVATIONS UNDER VARIOUS SCENARIOS:

The Snoop Protocol [13] was designed to solve two TCP unfriendly characteristics of wireless networks:

- Burst/intermittent packet loss due to high bit error rates
- Short temporary disconnections

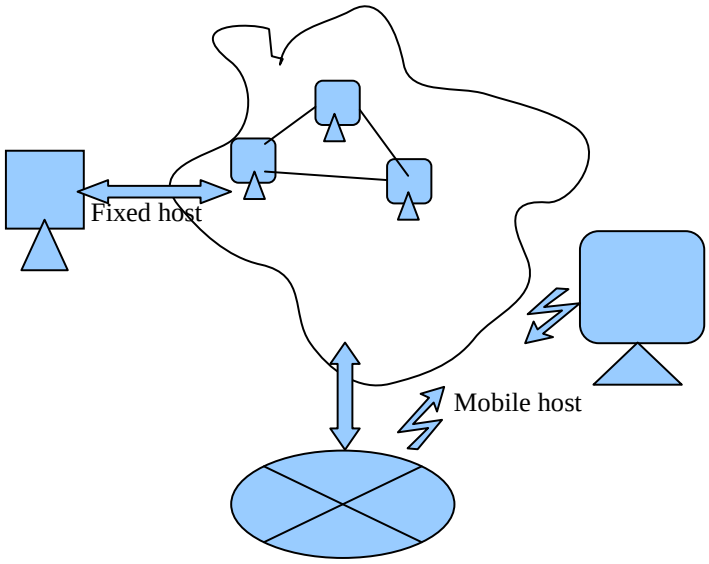

Figure4.1: Topology of a typical Snoop environment

Snoop shields the sender from the vagaries of the wireless link, without sacrificing the end-to-end semantics, or requiring any changes to the existing implementations of TCP. It, does not change or interfere with the content of the TCP packets that flow between the hosts. In wireless LANs, we expect to have administrative control over the last hop router, or base station (BS). The Snoop agent is designed to reside on the router between the wired and wireless networks, referred to as the gateway, or base station (BS) [Fig. 4.1]. Snoop is TCP aware, and using its knowledge of the congestion control mechanism in TCP along with its capability of identifying packet losses, Snoop performs local retransmission and recovery.

4.1Loading and Unloading functions of Snoop Module:

When the module is loaded, the snoop_conn_ init() function is run [Fig. 4.1.1].This function allocates space for the data structures that we have discussed above and prepares snoop for operation. When the module is unloaded, using snoop_clean _control() [Fig. 4.1.2], all the buffer space, all the buffered packets are thrown, connection states are emptied, and all the space allocated to the data structures during initialization is freed.

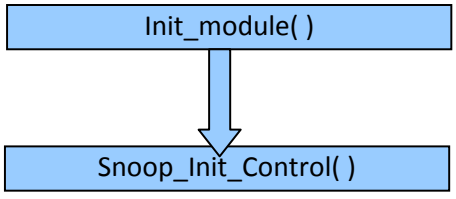

Figure 4.1.1: Loading the Module

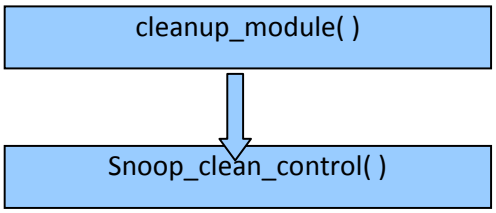

Figure 4.1.2: Unloading the Module

4.2 Proposed scenario:

The following scenario is created to study the behavior of snoop in Wired cum wireless network.

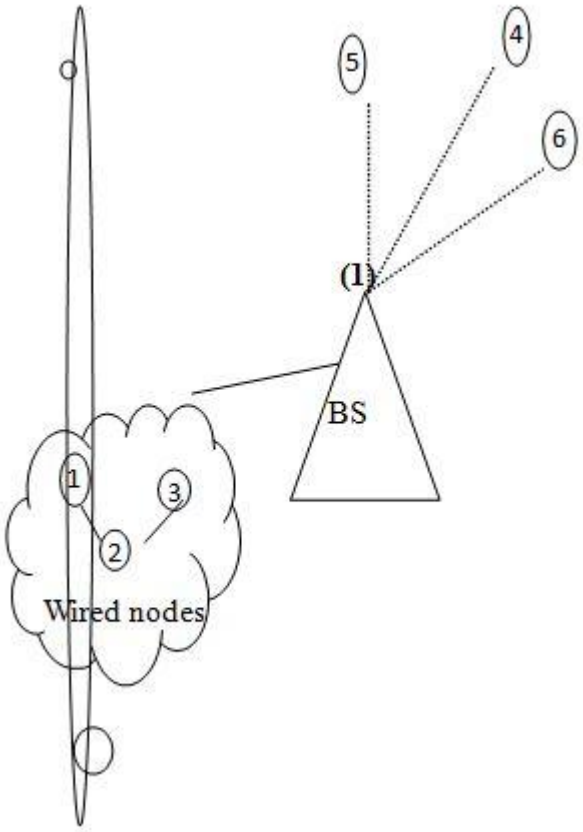

Fig4.2.1: Wired cum Wireless Scenario.

In above scenario, we used NS2 to perform the simulation of a wired cum wireless Scenario that consist of 6 nodes. The three nodes i.e. Nodes1, 2, 3 are wired network and node 4,5,6 are in wireless fashion. The base station lies between wired and wireless network. We applied Snoop protocol on this base station so that it monitors the performance of TCP in both environments. The parameters used were two way propagation model, 802.11 as the MAC, Omni directional antenna and drop tail interface queue with length of 50 for every node. We used Destination-Sequenced Distance-Vector Routing (DSDV) as the routing algorithm.

4.3 Flow Control and Congestion control

TCP, the reliable transport protocol was designed for traditional networks which consisted of wired links and stationary nodes. It performs very well on such networks adapting to end to end delays, and packet losses due to congestion at the intermediate routers. Once the connection is established, the each host advertises its receiver window size called the 'offered window'. It is the amount of receiving buffer available and it is the maximum amount of data the sender can send to the receiver at a time. As packets are received, they are stored in the receiver buffer until they are passed to the higher layer[15]. The receiver, advertises the size of its receiver window with each acknowledgement it sends to the sender. This causes the sender to not send excess data, and hence the receiver buffer never overflows. When the receiver's buffer is full, a window size of 0 is advertised, and the sender goes into the persist mode, where it stops sending data to the receiver without invoking any congestion control mechanism.

Slow start added another window to the sender's TCP called the congestion window (cwnd). It observes the congestion on the network and controls the amount of data that the sender pushes into the network. Upon connection establishment, cwnd starts from 1, and TCP increments cwnd by one for every ack that is received. The sender can transmit up to the minimum of the congestion and the advertised window[16]. The congestion window is therefore a

flow control imposed by the sender, where as the advertised window is by the receiver.

4.3.1 Implementing Congestion Avoidance Algorithm

Slow start is the way to initiate data flow across a connection. But at some point we'll reach the limit of an intervening router, and packets can be dropped. Congestion avoidance is a way to deal with lost packets[14] . It is described in [Jacobson 1988]. The assumption of the algorithm is that packet loss caused by damage is very small (much less than 1%), therefore the loss of a packet signals congestion somewhere in the network between the source and destination. There are two indications of packet loss: a timeout occurring and the receipt of duplicate ACKs. If we are using a timeout as an indication of congestion, we can see the need for a good RTT algorithm.

Congestion avoidance and slow start are independent algorithms with different objectives.. But when congestion occurs we want to slow down the transmission rate of packets into the network, and then invoke slow start to get things going again. In practice they are implemented together. Congestion avoidance and slow start require that two variables be maintained for each connection: a congestion window, cwnd, and a slow start threshold size, ssthresh[14] The combined algorithm operates as follows:

1.Initialization for a given connection sets cwnd to one segment and ssthresh to 65535 bytes.

2. The TCP output routine never sends more than the minimum of cwnd and the receiver's advertised window. Congestion avoidance is flow control imposed by the sender, while the advertised window is flow control imposed by the receiver. The former is based on the sender's assessment of perceived network congestion; the latter is related to the amount of available buffer space at the receiver for this connection.

3. When congestion occurs (indicated by a timeout or the reception of duplicate ACKs), one-half of the current window size (the minimum of cwnd and the receiver's advertised window, but at least two segments) is saved in ssthresh. Additionally, if the congestion is indicated by a timeout, cwnd is set to one segment (i.e., slow start).

4.When new data is acknowledged by the other end, we increase cwnd, but the way it increases depends on whether we're performing slow start or congestion avoidance.
If cwnd is less than or equal to ssthresh, we're doing slow start; otherwise we're doing congestion avoidance. Slow start continues until we're halfway to where we were when congestion occurred (since we recorded half of the window size that got us into trouble in step 2), and then congestion avoidance takes over.
We can see the values of *cwnd* and *ssthresh* as each segment is transmitted. If the MSS is 128 bytes, the initial values of *cwnd* and *ssthresh* are 128 and 65535, respectively. Each time an ACK is received we can see *cwnd* incremented by the MSS, taking on the values 512, 768, 1024, 1280, and so on. Assuming congestion doesn't occur, eventually the congestion window will exceed the receiver's advertised window, meaning the advertised window will limit the data flow.

A more interesting example is to see what happens when congestion occurs. There were four occurrences of congestion while this example was being run. There was a timeout on the transmission of the initial SYN to establish the connection, followed by three lost packets during the data transfer.

Figure 4.3 shows the values of the two variables *cwnd* and *ssthresh* when the initial SYN is retransmitted, followed by the first seven data segments.

We taken these values of cwnd by using following formula:

cwnd= cwnd+ (segsize*segsize)/cwnd+(segsize/8)

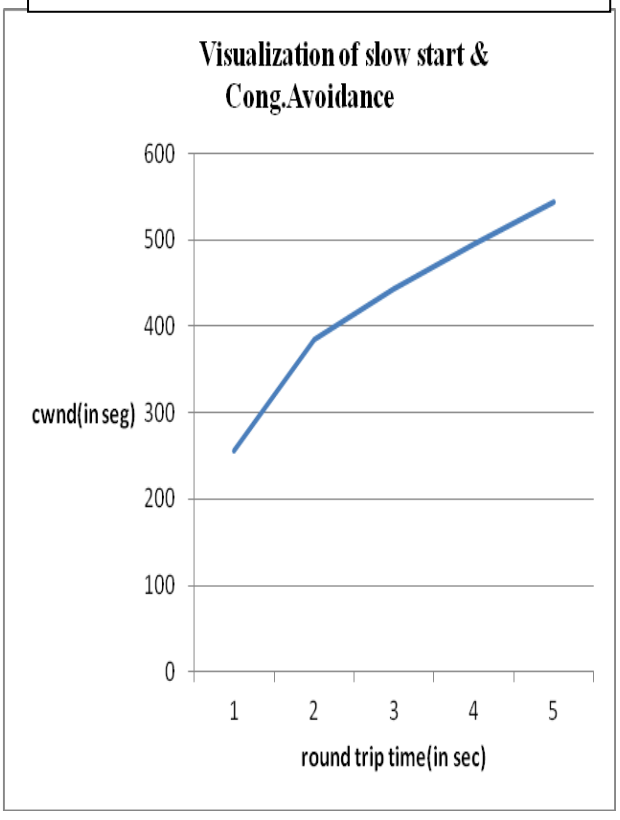

Figure 4..3: is a visual description of slow start and congestion avoidance. cwnd and ssthresh are maintained in bytes.

Table 4.1 Values of cwnd for Congestion Avoidance

Segment	Send	Receive	Comment	cwnd	Ssthresh
	SYN SYN ACK	SYN,ACK	Initialize Timeout retransmit	128 128	655 35 384
1	1:129(128)	ACK 129	Slow start	256	
2					384
3	129:257(128)				
4	257:385(128)				
5		ACK 257	Slow start	384	
6	385:513(128)				
7	513:641(128)			443	384
8		ACK 513	Cong. avoid		
9	641:769(128)		Cong. avoid	495	
10		ACK 641	Cong. avoid		
11	769:897(128)			544	384
12		ACK 769			
					384
					384

V. CONCLUSION

In this paper, we studied the previous work on snoop protocol discussed by many authors. By this deep study, we came to know the problems faced by TCP when it works in wireless networks. But Snoop protocol is a better solution for these problems. The most well known TCP-aware link layer recovery scheme is the Snoop Protocol. Snoop protocol introduces a Snoop agent at the base station, which monitors packets flowing in both directions. It maintains a cache of the packets, and whenever a packet loss is detected, it does a local recovery, and drops all duplicate acknowledgements. So this advantage of Snoop will help us in our proposed work.

VI. FUTURE WORK

We suggest for future the impact of packet loss due to multiple reasons that need further investigation. Especially, when it is under same adversity or an attack. Because we do not know how snoop will behave when it comes under attack and same security is compromised. Therefore, this aspect is also need to take care for further future scope. In this paper we attempted to reduce some congestion by using Congestion avoidance algorithm but in future further work can be done to increase the performance of wired cum wireless network.

REFERENCES

[1] Kostas Pentikousis "*TCP in wired-cum-wireless environments"* Department of Computer Science State Uiversity of New York at Stony Brook.
[2] "I-TCP: Indirect TCP for mobile hosts," in Proc. 15th Int. Conf. Distributed Computing Syst. (ICDCS), May 1995[3]
[3] H . Balakrishnan, S. Seshan, and R. H. Katz, "Improving reliable transport and handoff performance in cellular wireless networks," ACM Wireless Networks, vol. 1, Dec. 1995.
[4] R. Yavatkar and N. Bhagwat, "Improving end-to-end performance of TCP over mobile internetworks," in Mobile 94 Workshop Mobile Computing Syst. Appl., Dec. 1994.
[5] E. Ayanoglu, S. Paul, T. F. LaPorta, K. K. Sabnani, and R. D. Gitlin, "AIRMAIL: A link-layer protocol for wireless networks," ACM ACM/Baltzer Wireless Networks J., vol. 1, pp. 47–60, Feb. 1995.
[6] Ashish Natani , et.al "TCP for Wireless Networks" Computer Science Program, University of Texas at Dallas, Richardson, November 12, 2001.
[7] Dimitrios Koutsonikolas, et.al" lOn TCP Throughput and Window Size in Multihop Wireless Network "Testbed ,Center for Wireless Systems and Applications, Purdue University.
[8] Prasad Nambiar ,et.al "Snoop Behaviour in Multihop Wireless Networks "School of Computer Science University of Hertfordshire Hatfield Hertfordshire,2010.

[9] Mr. Manish, D.Chawhan, Dr Avichal R.Kapur "Performance Enhancement of TCP Using ECN and Snoop Protocol for Wi-Max Network" , Shri Ramdeobaba Kamla Nehru College of Engg, International Journal of Computer Applications.

[10] Srikanth Tiyyagura ,Rajesh Nutangi ”An Improved Snoop For TCP RENO And TCP SACK In Wired-Cum-Wireless Networks.”Department of Computer Science and Engineering JNTUA College of Engg., pulivendula , Andhra Pradesh, India. july ,2011.

[11] M. Alnuem, J. Mellor ”TCP Multiple drop action for transmission errors ”School of Informatics, University of Bradford ,2008.

[12] In Huh ,et.al ”Decision of Maximum Congestion Window Size for TCP Performance Improvement by Bandwidth and RTT Measurement in Wireless Multi-Hop Networks”, International Journal of Information Processing Systems, Vol.2, No.1, March 2006

[13] Amir, E., Balakrishnan, H., Seshan S. and Katz, R. H. Efficient TCP over networks with wireless links, IEEE, September 1994.

[14] W. Richard Stevens, G. Gabrani ”TCP/IP Illustrated, The Protocols Volume 1”published by Pearson Education,2009.

[15] Parminder Singh "Performance Issues and Comparative Study between TCP over Wireless Link Approaches",ICACCT,2012.

[16] Parminder Singh , Kanwalvir Dhindsa" Analytical study of performance of TCP Reno over wireless networks", ICETEC,2009.