AN IMPROVED SNOOP FOR TCP RENO AND TCP SACK IN WIRED-CUM-WIRELESS NETWORKS

Srikanth Tiyyagura

Department of Computer Science and Engineering

JNTUA College of Engg., pulivendula,

Andhra Pradesh, India.

Rajesh Nutangi

Department of Computer Science and Engineering

JNTUA College of Engg., pulivendula,

Andhra Pradesh, India.

Dr. P. Chenna Reddy

Department of Computer Science and Engineering

JNTUA College of Engg., pulivendula,

Andhra Pradesh, India.

Abstract

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. This paper analyzes the distinct behavior and introduces a modification to the Snoop protocol for wired-cum-wireless networks. The modified snoop protocol improves the performance of TCP SACK by around 10% compared to the plain TCP Sack protocol and 5% in an environment where no TCP enhancing mechanism is in place. The performance of TCP RENO is also improved by 30% compared to the plain TCP Reno protocol and about 20% with no TCP enhancing mechanism used.

***Keywords*:** Snoop Protocol; wired-cum-wireless networks; TCP Variants.

1. Introduction

The reason for the poor performance of TCP in wireless networks is the implicit assumption that all packet losses are caused by network congestion. Losses in wireless networks can result from bit errors, fading and handoffs. Due to this routing of traffic from wired networks to wireless networks is a problem. To address this problem, many modifications and new solutions have been proposed to improve TCP performance, such as forward error correction schemes, retransmissions at the link layer, split connections, modifications to the TCP and so on.

The Snoop Protocol was designed to solve the burst/intermittent packet loss experienced by TCP in wireless link. The Snoop agent is designed to reside on the router between the wired and wireless link, referred to as the gateway, or base station. The role of the snoop agent is to monitor the TCP packets transmitted from a fixed host to a mobile host and vice versa. The agent caches all those packets locally and in the case of receiving duplicate acknowledgments, snoop retransmits the packets promptly from local cache and suppresses duplicate acknowledgements. Thus, the base station hides the packet loss from the fixed host by not propagating duplicate acknowledgments, thereby preventing unnecessary congestion control mechanism invocation.

In practice Snoop is not used in isolation and is combined with TCP variants. TCP Reno is the widely used TCP variant and TCP Sack is suitable for wired-cum-wireless networks. All the TCP variants are based on four algorithms Slowstart, Congestion avoidance, Fast Retransmit and Fast Recovery. But they differ in the way they implement these algorithms.

TCP Reno detects the lost packets earlier by using duplicate acknowledgements and it doesn’t take drastic measures when a packet is lost. The receiver sends a duplicate acknowledgement whenever it receives a packet out of order. The sender instead of waiting for retransmission time out waits for 3 duplicate acknowledgement. When sender receives 3 duplicate acknowledgements it retransmits what appears to be the missing packet. Thus Reno retransmits early resulting in improvement in performance. Also Reno doesn’t set the congestion window to 1 every time a packet is lost. It sets it to slow start threshold value.

TCP SACK is quite different from other TCP variants. Instead of using cumulative acknowledgement it uses selective acknowledgement. This modification was basically done for handling multiple packet losses in a window of data. A SACK receiver is able to exactly indicate the sender using Selective Acknowledgments which packets in a sequence of data have been received correctly and which have been not. This is done using a special type of Selective Acknowledgment segment called SACK block. A SACK block provides the sender with all the necessary information needed to retransmit the exact packets that are missing. It is able to thus efficiently cope with packet losses in the wireless channel and retransmit all the missing packets in one RTT, hence reducing TCP timeouts.

2. Related Work

The challenges that must be met in order to provide reliable transport services to all hosts regardless of the type of network connectivity used is studied in [1]. It surveys recently proposed solutions and evaluates them with respect to a wired-cum-wireless environment.

In [2] the progress in Various TCP Variants is studied and the performance of different TCP variants Tahoe, Reno, New Reno, Westwood, Selective Acknowledgment, Forward Acknowledgement and Vegas are studied. It is concluded that congestion is the major factor influencing the performance of various TCP variants.

In [3] popular mechanisms that have been proposed to solve the wireless transport problem are classified according to their complexity, ease of deployment, and performance in different scenarios. It is found that some TCP sender adaptations work well for certain environments only.

Simulations are used to explore the benefits of adding selective acknowledgments (SACK) and selective repeat to TCP [4]. Original version of Tahoe and Reno are compared, with the modified version of Reno with SACK option. It is observed that without selective acknowledgments, TCP variants are constrained to either retransmit at most one dropped packet per round-trip time, or to retransmit packets that might have already been successfully delivered.

The problems of wireless media are addressed [5] and solutions are designed to them. These solutions incorporate link-layer techniques as well as enhancements to TCP at the sender and receiver. The adverse effects of wireless bit-errors on TCP performance arise primarily because TCP misinterprets wireless losses as being due to congestion. The Berkeley Snoop Protocol was designed and implemented as a novel protocol that exploits cross-layer protocol optimizations to improve performance.

In [6] it is shown how Explicit Congestion Notification (ECN), Snoop protocol and their combination can be used to improve the performance of TCP in Wi-Fi Bidirectional Network and in Handoff. 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 avoids the decrease in transmission speed.

A simulation-based performance analysis of the most important TCP variants over wireless networks is included [7]. In addition, TCP versions in the same environment are analyzed but including the Snoop protocol, it is found that TCP Vegas and SACK present an opposite behavior.

The distinct behavior with TCP Sack and Snoop is analyzed and introduce the TCP SACK Aware Snoop protocol for wired-cum-wireless networks [8]. It is showed how to make the Snoop protocol TCP SACK-Aware and why using the Snoop protocol is in fact worse than not using any mechanism at all. The new mechanism improves the performance of TCP SACK by about 30% compared with the presence of the normal Snoop protocol and by about 8% in the case where no performance improvement mechanism is used.

3. Simulation

3.1. *Environment*

Simulation environment is a wired-cum-wireless network implemented in ns-2 simulator. In wired network, there are 8 wired nodes and two 802.3 LAN Nodes where Snoop protocol is implemented. In wireless environment, there are 4 nodes and one base station. All wired nodes are connected with a 10Mbps, 10msec delay with Drop Tail bidirectional link. 802.3 LAN is created with 10Mbps, 10msec delay with Drop Tail queue. Snoop protocol is implemented in the LAN Node.LAN Node is connected to the base station with 500 Kbps, 10msec delay and RED bidirectional link. Red link is for marking the packets in case congestion occurs.

The wired node (5) sends the data packets to the wireless node (15), the protocol used is TCP SACK or TCP Reno. It is attached with a Constant Bit Generator (CBR) Traffic Generator. The wired node (0) sends the data packets to the wireless node (7), which is a wired to wired data transmission path. The wireless node (14) sends data to the wireless node (16), which constitutes the wireless to wireless data transmission path. The links 0 to 7 and 14 to 16 are created as side traffic.

The protocols used in the simulation are TCP Reno, TCP Sack and TCP Vegas. The other mechanisms used are Explicit Congestion Notification (ECN), Snoop and the modified snoop. The wireless routing protocol used is DSDV (Destination Sequence Distance Vector). All results are converted into graphs using x-graph tool.

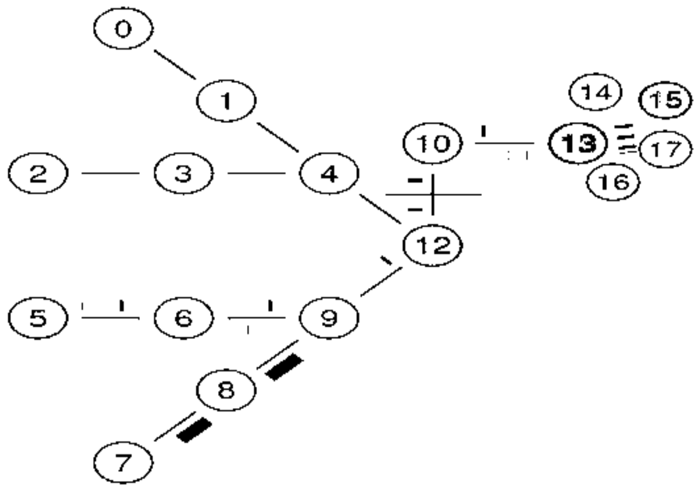

Fig 1: Wired-cum-wireless Scenario

3.2. *Evaluation Criteria*

The performance parameters which are used for evaluation are as follows

(1) Throughput: number of bits transmitted per second.

(2) Packet Loss: total number of packets dropped in the network.
(3) Congestion Window Size: used to estimate the congestion in the network.
(4) Packet Sequence Number: used to calculate total number of packets sent

3.3. *Simulation Results and Analysis*

In the simulated wired-cum-wireless environment, TCP Reno is used as transport protocol and Snoop protocol is enabled. The values of throughput, packet sequence number, and congestion window size and packet loss in wired and wireless networks are drawn. Next the TCP Sack is implemented as transport protocol along with snoop and then finally TCP Vegas is implemented with snoop and all the results are compared.

In the fig 2, the throughput vs. Time of TCP variants along with snoop are compared and the conclusion drawn is that the TCP Vegas , which is the least performing one in wired-cum-wireless networks without snoop become the best performing one and TCP Sack which is the best without snoop gives low performance in the presence of snoop. The TCP Reno is also giving low performance in the presence of snoop protocol. However, the throughput drawn by TCP Sack in the absence of snoop is much more than the throughput drawn by TCP Vegas in the presence of snoop.

In the fig 3, packet sequence number vs. Time graph of TCP Variants are compared and the maxim sequence number of the packets is achieved by TCP Vegas, followed by TCP Reno and finally TCP Sack. Theoretically, whenever snoop is added to the TCP, the performance of the TCP should increase but in the case of TCP Sack and TCP Reno, this is proved to be wrong.

Whenever a packet is lost, The TCP-Reno receiver sends the duplicate acknowledgements to the source. The TCP Reno source by using the new fast recovery algorithm, count the number of duplicate acknowledgements and increases the congestion window. This change will improve the throughput. But due to the snoop interference, the duplicate acknowledgements are dropped which affect the TCP Reno receiver's congestion window size. By using snoop, the new fast recovery algorithm is useless, which gives the low throughput.

Whenever packet burst occurs, the TCP Sack receiver sends a sack duplicate acknowledgements to the source providing the information of the lost packets and also the packets that are received. Then TCP SACK source will retransmit all the lost packets in a single RTT. But due to Snoop's interference only one packet is retransmitted and the duplicate ACK's (SACK ACK's) are dropped. The SACK sender is thus not able to retransmit all the packets in one RTT and the basic SACK mechanism is thus disrupted. Therefore the TCP Sack throughput is decreasing when used with snoop.

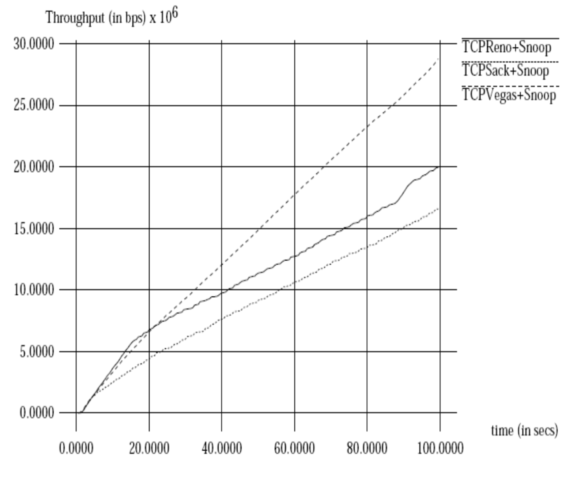

Fig 2: Throughput vs. Time at the Mobile Node

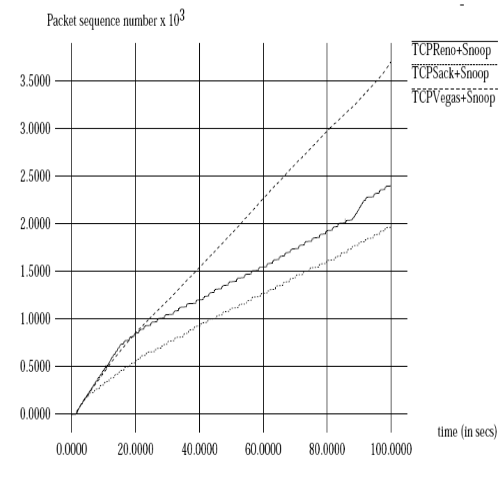

Fig 3: Packet Sequence number vs. Time for TCP variants

The solution for solving this problem is "No need to drop duplicate acknowledgements, just forward them to the sender". Now, sender can use their own processing's and in addition snoop helps as a buffering agent and retransmits the local stored packet. The sender also sends the same packet but it may undergo loss or anything may happen to the packet in mean while. Overall, it solves both TCP Reno and also TCP Sack problem in increasing the throughput of the network.

In the Fig 4, Throughput vs. Time graph for TCP Reno with snoop in one case and with modified snoop in another case is drawn and the maximum throughput is achieved by TCP Reno + modified Snoop followed by TCP Reno + Snoop. In the Fig 5, packet sequence number vs. Time graph for TCP Reno is drawn and the maxim sequence number is achieved by TCP Reno + modified snoop, followed by TCP Reno+ snoop.

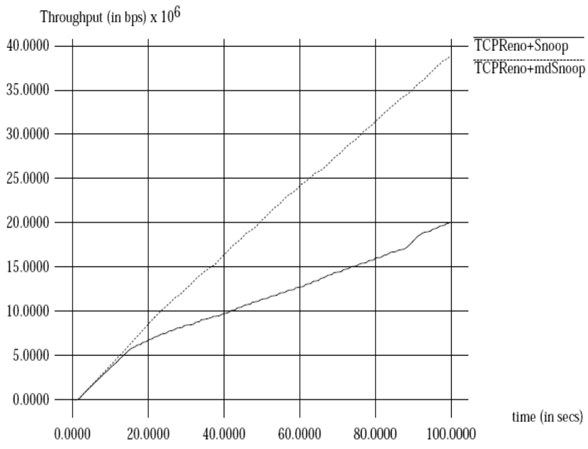

Fig 4: Throughput vs. Time for TCP Reno + modified snoop

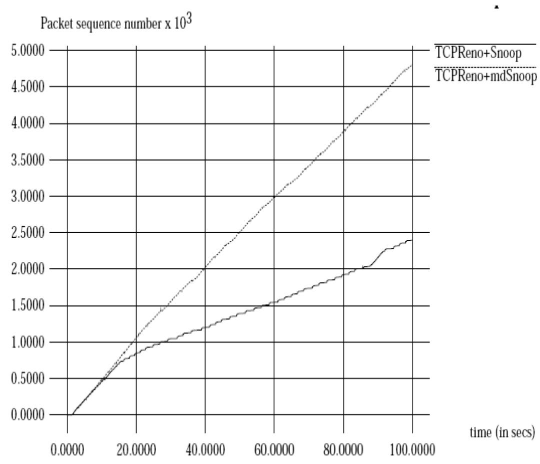

Fig 5: Packet sequence number vs. Time for TCP Reno + modified snoop

In the Fig 6, Throughput vs. Time graph for TCP Sack is drawn and the maximum throughput is achieved by TCP Sack + modified Snoop followed by TCP Sack + Snoop. In the Fig 7, packet sequence number vs. Time graph for TCP Sack is drawn and the maxim sequence number is achieved by TCP Sack + modified snoop, followed by TCP Sack+ snoop

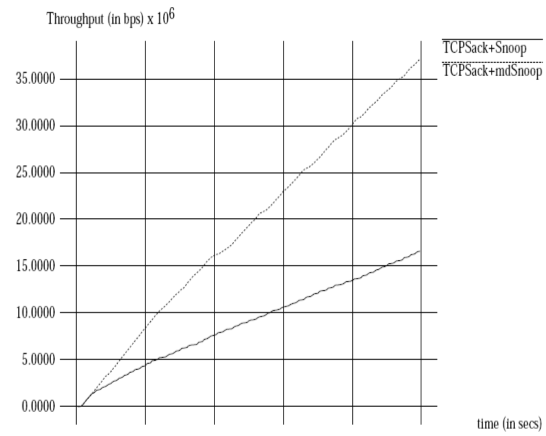

Fig 6: Throughput vs. Time for TCP Sack+ modified snoop

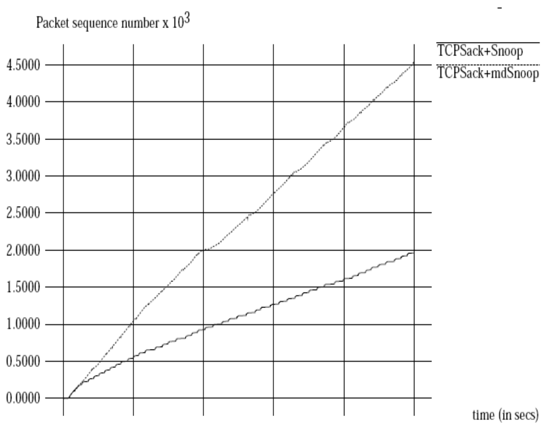

Fig 7: Packet Sequence number vs. Time for TCP Sack+ modified snoop

Table 1. Over-view of throughput values in bps at Base station

TCP VARIANTS	GENERAL	SNOOP	Modified snoop	Modified Snoop + ECN
TCP Sack	359644	166203	372891	382558
TCP Reno	326742	200123	389205	416301

TCP Vegas	137485	289801	----	----

4. Conclusion

This paper analyzes the behavior of TCP SACK and TCP Reno in the presence of the Snoop protocol, one of the best known solutions for improving TCP's performance over wireless networks. 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.

References

[1] Kostas Pentikoussis. "TCP in Wired-cum-Wireless Environments". IEEE Communications Surveys, Vol. 3, No. 4:2–14, 2000.
[2] B. Qureshi, M. Othman, Member, IEEE, and N. A. W. Hamid. "Progress in Various TCP Variants (February 2009)". Computer, Control and Communication, 2009. 17-18 Feb. 2009 :1 - 6
[3] Sonia Fahmy, Venkatesh Prabhakar, Srinivas R. Avasarala, Ossama M. Younis. "TCP over Wireless Links: Mechanisms and Implications". CSD-TR-03-004, Purdue University 2003.
[4] S. Floyd and K. Fall. "Simulation Based Comparisons of Tahoe, Reno and Sack TCP". ACM Computer Communication Review, 26, No. 3:5–21, 1996.
[5] Balakrishnan, V. N. Padmanabhan, S. Seshan, and R. H. Katz, "A comparison of mechanisms for improving TCP performance over wireless links," IEEE/ACM Transactions on Networking 5(6), pp. 756– 769, 1997.
[6] Manish Devendra Chawhan1 Dr Avichal R.Kapur2." TCP Performance Enhancement Using ECN And Snoop Protocol for Wi-Fi Network". Issue : April 2010, pp. 186-190
[7] S. Vangala and Miguel A. Labrador. "Performance of TCP over Wireless Networks with the Snoop Protocol". In Proceedings of IEEE LCN, pages 600–601, Tampa, FL, November 2002.
[8] Sarma Vangala and Miguel A. Labrador. "The TCP SACK-Aware Snoop Protocol for TCP over Wireless Networks". Proceedings of IEEE VTC 2003, pp. 2624-2628, Orlando, October 2003
[9] L. S. Brakmo, S. W. O'Malley, and L. L. Peterson. "TCP Vegas: New Techniques for Congestion Detection and Avoidance". In Proceedings of SIGCOMM, pages 24–35, 1994.
[10] A. Bakre and B.R. Badrinath. "I-TCP: Indirect TCP for Mobile Hosts". In Proceedings of ICDCS, pages 136–143, 1995.
[11] B. S. Bakshi, N. Vaidya, and D. K. Pradhan. "Improving the Performance of TCP over Wireless Networks". In Proceedings of 17th. International Conference on DCS, pages 365–373, July 1997.
[12] A. DeSimone, M. C. Chuah, and O. C. Yue. "Throughput Performance of Transport Layer Protocols over Wireless LANs". In Proceedings of GLOBECOM, pages 36–46, 1993.
[13] C. Parsa and J. J. Garcia-Luna-Aceves. "Improving TCP Performance over Wireless Networks at the Link Layer". Mobile Networks and Applications, 5(1):57–71, 2000.