LETTER

# Improved Transport Layer Performance Enhancing Proxy for Wireless Networks

**Jeng-Ji HUANG**[†a)] *and* **Huei-Wen FERNG**[††], ***Nonmembers***

**SUMMARY** It is well known that deploying a proxy at the boundary of wireless networks and the Internet is able to improve the performance of transmission control protocol (TCP) over wireless links. Snoop protocol, acting like a transport layer proxy, performs local retransmissions for packets corrupted by wireless channel errors. In this letter, an improvement for the Snoop protocol is proposed to shorten the time spent on local recovery by sending extra copies in every local retransmission attempt. This enables TCP to quickly return to normal, effectively eliminating several of the problems that may cause throughput degradation.

***key words:*** *TCP, wireless networks, performance enhancing proxy, throughput performance, Snoop protocol*

## 1. Introduction

Communication over wireless links inevitably incurs sporadic high bit-error rates. TCP, which is currently the most widely used end-to-end transport protocol on the Internet, provides reliable data transfer between two hosts. It is tuned to perform well in traditional wired networks; however, when packet losses occur due to corruption in networks with wireless links, TCP misjudges them as sign of network congestion and triggers congestion control and avoidance mechanisms, resulting in unnecessary reduction in end-to-end throughput. Figure 1 shows the system architecture.

To improve the performance of TCP over wireless links, a transport layer performance enhancing proxy (PEP), e.g., [1], [2], can be placed at the border of wireless networks and the Internet; the end-to-end semantics is still preserved if TCP is not split at the PEP. Snoop protocol [3], [4] can be viewed as one such a PEP that aids to increase the robustness of data delivery across wireless links by performing local retransmissions for packets damaged by wireless channel errors. It caches packets sent across the wireless link, performs local retransmission after the arrival of a small number of duplicate acknowledgments (ACKs) from the receiver or after a local timeout, and suppresses the duplicate ACKs. Figure 2 shows the protocol layer of a data connection between the server and a mobile host.

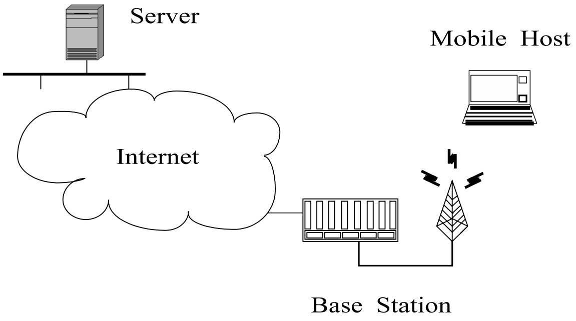

**Fig. 1** The system architecture.

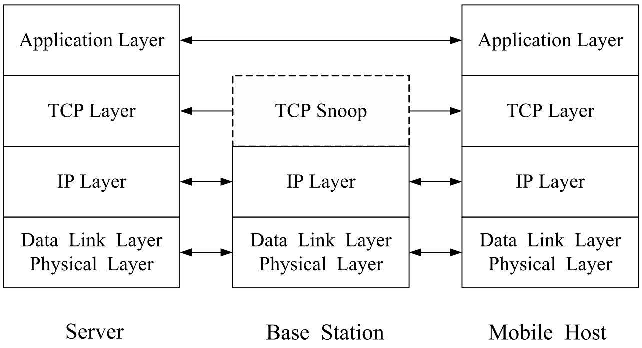

**Fig. 2** The protocol layer of a data connection between the server and a mobile host.

Although previous studies have shown that Snoop significantly improves the performance of TCP in networks containing wireless links [3], [4], there still exist three problems that may degrade the end-to-end throughput; i.e., TCP stalls while local retransmission is performed; the problem of creating a burst of traffic is introduced as multiple packets are acked by a single new ACK after local retransmission is performed successfully [5]; and a TCP spurious timeout may occur when too many local retransmissions are needed for a packet before successfully received by the mobile host.

In order to mitigate these problems, Snoop suggests retransmitting packets at a higher priority* [3]. This enables retransmitted packets to reach the mobile host sooner, reducing the number of duplicate ACKs and leading to improved throughput. However, as will be seen in Sect. 3, the gain coming from this is only marginal. On the other hand, Eifel [7], F-RTO [8], or STD [2] are proposals aiming at detecting spurious timeout and responding to it by either reversing the congestion control state [7], [8] or filtering duplicate ACKs that can cause spurious fast retransmission[†] [2]. However,

---

Manuscript received May 12, 2005.

[†]The author is with the Department of Industrial Education, National Taiwan Normal University, Taipei 106, Taiwan.

[††]The author is with the Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology, Taipei 106, Taiwan.

a) E-mail: hjj2005@ntnu.edu.tw

DOI: 10.1093/ietcom/e89–b.1.206

[*]This function is originally not implemented in ns-2 [6], but is added in this letter for the purpose of comparison in Sect. 3.

all of them passively detect spurious timeouts rather than actively suppress them. In this letter, an improvement of the Snoop protocol, called Snoop+, is proposed to help TCP to quickly return to normal. This effectively alleviates all the problems mentioned previously and it is done by only a small change at Snoop's retransmission mechanism.

## 2. Snoop+

In Snoop+, instead of retransmitting one copy at a time, multiple copies are sent over the wireless link in every local retransmission attempt. Obviously, increasing the number of copies retransmitted would raise the chance of successful transmission of at least one copy, resulting in a faster local recovery. However, as explained below, this may not always be true and this number can not be increased without bound especially when the wireless link is in good channel conditions.

In Snoop, local retransmission is triggered when the number of duplicate ACKs seen at the base station (BS) reaches a retransmission threshold, RTX_THRESH, which is typically set to 3 in order to account for possible reordering of TCP packets[††]. However, when no packet reordering occurs, in Snoop+ extra copies are added into every local retransmission trial. As will be seen in Sect. 3, this small change in the number of copies retransmitted leads to a significant improvement of the end-to-end throughput. It should be noted that when the number of extra copies are larger than or equal to 3, another local retransmission may be falsely triggered. For example, if 3 extra copies are used in Snoop+ and when all copies are correctly received by the mobile host, a new ACK and 3 duplicate ACKs are then generated upon the reception of the original copy and the following 3 extra copies, respectively. These 3 duplicate ACKs will eventually be collected at the BS and an unnecessary local retransmission will as a result be falsely triggered.

Multiple-copy transmission has already been widely discussed in designing ARQ protocols [9], [10], but not yet been considered at transport layer due to the concern of network efficiency or goodput. This letter exploits the possibility of performing transport layer multiple-copy transmission over wireless links, i.e., locally at a PEP, and shows its advantage in improving the end-to-end throughput of TCP.

## 3. Simulation Results and Discussions

The throughput performance of Snoop and Snoop+ is compared through simulations run on ns-2 [6], and TCP Reno is used. A topology consisting of a wired link and a wireless link is simulated, where the propagation delay and the bandwidth is 30 msec and 10 Mbps for the wired link, and 1 msec and 2 Mbps for the wireless link, respectively. The wireless channel errors are characterized by either random errors or correlated errors. Each simulation is run for 1,000 s to get a stable result; TCP packet size is 512 bytes; and ACK size is 40 bytes. If not specifically mentioned, retransmission is given a higher priority in both schemes. Packet reordering is not considered in order to see the maximal performance gain that can be achieved from using the proposed scheme. In the following, for simplicity, Snoop+ using $c$ extra copies is denoted by Snoop+/$c$. Therefore, Snoop is equivalent to Snoop+/0, as no extra copy is sent.

### 3.1 Random Errors

In this section, we examine the performance of two schemes under the assumption that packet loss rate (PLR) on the wireless link is uniformly distributed [4], [5], [11]. First of all, the throughput performance of Snoop is compared between when priority is given to retransmitted packets (denoted by Snoop) and when no priority is used (denoted by Snoop without priority). As can be seen in Fig. 3, Snoop performs better when retransmitted packets are given precedence; however, the gain is slight, showing that simply giving priority to retransmitted packets has only limited effect on improving the end-to-end throughput of TCP.

Next, throughput performance is compared between Snoop and Snoop+ under various numbers of extra copies. As shown also in Fig. 3, Snoop+/1 and Snoop+/2 substantially outperform Snoop over a wide range of wireless channel conditions. Furthermore, Snoop+/1 performs slightly better than Snoop+/2 when PLR is low (smaller than 3%), but the situation is reversed when PLR is high. On the other hand, the throughput of Snoop+/3 degrades severely, especially when PLR is low. It is because a lot of unnecessary local retransmissions are falsely triggered when all copies are received correctly at the mobile host as mentioned in Sect. 2. Generally speaking, Snoop+/2 performs the best; its improvement over Snoop grows as PLR increases and can exceed 200%.

### 3.2 Correlated Errors

In this section, the wireless channel errors are modeled as a two-state, a good state and a bad state, Markov process [12]. The residual time in either state is exponentially distributed. Packet transmissions are assumed to be successful in a good state and corrupted in a bad state, with probability both equal to one. The mean good state period is fixed at 100 msec, while the mean bad state period is varied from 1 to 10 msec.

As can be seen in Fig. 4, comparing between Snoop and Snoop without priority, the former performs better but only slightly, similar to the result obtained previously under random channel errors. On the other hand, Snoop+/1 and Snoop+/2 significantly improve the throughput performance of Snoop over a wide range of channel conditions, although the performance gain shrinks due to that all retransmitted

---

[†] It should be noted that prevention of redundant transmission has already been included in Snoop as one of its basic functions; thus, the problem of spurious fast retransmission can be completely avoided, even in the absence of [2].

[††] In ns-2, RTX_THRESH is 1 by default since no reordering would occur.

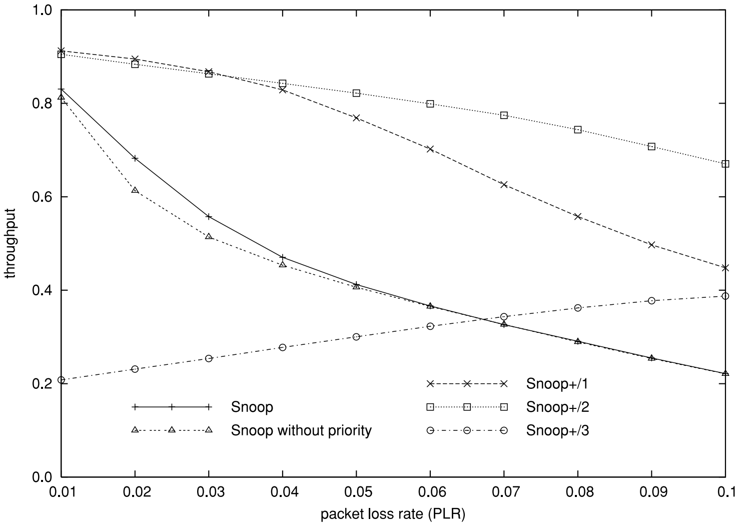

**Fig. 3** Throughput performance comparison under random channel errors.

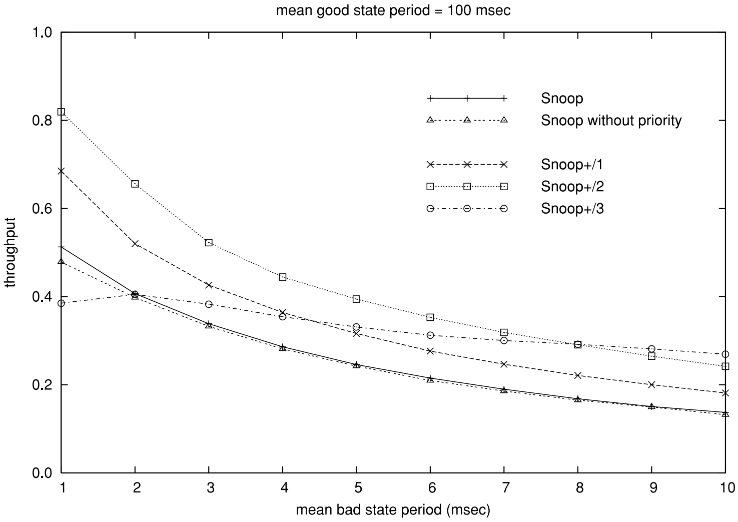

**Fig. 4** Throughput performance comparison under correlated channel errors.

packets, including both original and extra copies, could be damaged during a channel bad state. In general, Snoop+/2 still has the best throughput performance with improvement over Snoop ranging from 60% to 76%. It should be noted that when mean bad state period is small, as expected, the throughput of Snoop+/3 degrades severely; but, when mean bad state period is larger than or equal to 8 msec Snoop+/3 performs even better than Snoop+/2 with improvement up to 96% over Snoop. Based on Figs. 3 and 4, it is obvious that the throughput performance of Snoop+ can therefore be optimized by dynamically adjusting the number of extra copies according to the wireless channel condition.

## 4. Conclusions

In this letter, an improved Snoop, called Snoop+, is proposed to reduce the time spent on local recovery and thereby to enhance the end-to-end throughput of TCP via adding extra copies into every local retransmission attempt. Through extensive simulations, the following important results have been obtained. First, retransmission with a higher priority, as suggested in Snoop to speed up local recovery, offers only a limited improvement in performance. Second, Snoop+ significantly improves the end-to-end throughput of Snoop over a wide range of wireless channel conditions; the improvement can exceed 200% under random channel errors and can be up to 96% under correlated channel errors. Finally, Snoop+ with 2 extra copies overall yields the best throughput; and the optimal throughput performance of Snoop+ can be obtained by dynamically adjusting the number of extra copies according to the wireless channel condition.

## References

[1] M. Meyer, J. Sachs, and M. Holzke, "Performance evaluation of a

TCP proxy in WCDMA networks," IEEE Wirel. Commun., vol.10, no.5, pp.70–79, Oct. 2003.

[2] Y.C. Kim and D.H. Cho, "Considering spurious timeout in proxy for improving TCP performance in wireless networks," IEEE Commun. Lett., vol.8, no.1, pp.30–32, Jan. 2004.

[3] H. Balakrishnan, S. Seshan, and R.H. Katz, "Improving reliable transport and handoff performance in cellular wireless networks," Wirel. Netw., vol.1, no.4, pp.469–481, Dec. 1995.

[4] J.H. Hu, G. Feng, and K.L. Yeung, "Hierarchical cache design for enhancing TCP over heterogeneous networks with wired and wireless links," IEEE Trans. Wirel. Commun., vol.2, no.2, pp.205–217, March 2003.

[5] J.H. Hu and K.L. Yeung, "FDA: A novel base station flow control scheme for TCP over heterogeneous networks," Proc. IEEE INFOCOM '01, pp.142–151, 2001.

[6] S. McCanne and S. Floyd, "ns (network simulator)," http://www.isi.edu/nsnam/ns

[7] R. Ludwig and R.H. Katz, "The Eifel algorithm: Making TCP robust against spurious retransmissions," ACM SIGCOMM '00, pp.30–36, Jan. 2000.

[8] P. Sarolahti, M. Kojo, and K. Raatikainen, "F-RTO: A new recovery algorithm for TCP retransmission timeouts," Technical Report C-2002-07, Department of Computer Science, University of Helsinki, Feb. 2002.

[9] J. Zhu and S. Roy, "Improving link layer performance on satellite channels with shadowing via delayed two-copy selective repeat ARQ," IEEE J. Sel. Areas Commun., vol.22, no.3, pp.472–481, April 2004.

[10] W.S. Jeon and D.G. Jeong, "Improved selective repeat ARQ scheme for mobile multimedia communications," IEEE Commun. Lett., vol.4, no.2, pp.46–48, Feb. 2000.

[11] C.P. Fu and S.C. Liew, "TCP veno: TCP enhancement for transmission over wireless access networks," IEEE J. Sel. Areas Commun., vol.21, no.2, pp.216–228, Feb. 2003.

[12] M. Zorzi, A. Chockalingam, and R.R. Rao, "Throughput analysis of TCP on channels with memory," IEEE J. Sel. Areas Commun., vol.7, no.4, pp.458–472, Aug. 1999.