Evaluation of Mitigation to Bursty Packets by a TCP Proxy in a Wired and Wireless Network

Toshihiro Shikama†

†Fukui University of Technology, Japan
shikama@fukui-ut.ac.jp

Abstract - This paper investigates a TCP proxy that splits a TCP connection into two parts consisting of a wireless link and a wired network. The TCP proxy is effective for improving TCP performance in such a heterogeneous network including a wireless link, on which transmission errors occur. This paper describes how the TCP proxy produces large size forward data due to a packet loss on the wireless link. It also identifies a new problem that the output traffic from the TCP proxy becomes bursty due to the large size forward data. To mitigate bursts of packets by the TCP proxy this paper proposes a scheme that the proxy performs a pacing function, which places a gap between two consecutive packets. Since this function is performed in cooperation with the flow control between two TCP connections, the scheme has an advantage that it requires a small amount of forwarding buffers for the pacing. Simulation results using ns-2 show that bursty packets are produced by the conventional TCP proxy and the pacing function suppresses them. The results also show that throughput is improved by the proposed scheme, while the throughput of the conventional TCP proxy suffers from packet losses in the wired network due to the bursts of packets.

Keywords: TCP proxy, PEP, burst, pacing, reassembling

1 INTRODUCTION

Transmission errors have to be considered in IP networks that employ wireless links. Although TCP is mainly employed end-to-end in the Internet, it is well understood that TCP cannot achieve sufficient throughput in the environment where packets are lost due to transmission errors. TCP assumes that losses of packets are derived from a buffer overflow at a forwarding node; it invokes the congestion control to reduce traffic. In the case where losses of packets are caused by transmission errors on a wireless link, the congestion control is performed unnecessarily; a degradation of TCP throughput occurs.

One approach to mitigate this problem is to employ a TCP proxy called PEP (Performance Enhancing Proxy) that terminates a TCP connection from a source terminal and establishes another TCP connection to the destination terminal [1]. It forwards receive data from one TCP connection to another. As the large round-trip delay causes low throughput and the PEP makes the delay short by splitting a TCP connection, the throughput is improved. Although termination of a TCP connection by the PEP has a significant impact on the characteristics of an IP flow, a study on this aspect hitherto has not been done. This paper focuses on the property of output traffic from the PEP and identifies a problem concerning bursts of packets due to the termination of a TCP connection. It also proposes a mitigation scheme for the problem and evaluates its effectiveness by simulations using ns-2.

This paper assumes that a PEP is placed in a node which connects a wireless link to a wired network. In the following description, we introduce the term “a wireless section” to refer to the wireless link. We also call the wired network “a wired section.” Figure 1 illustrates an example of an error recovery sequence by the PEP placed between the wireless section and the wired section. When a loss of a packet occurs on the wireless section, an error recovery by a retransmission of associated segment is performed by TCP over the wireless section. The figure assumes that the fast retransmit is invoked after three duplicate ACKs. If a receiving side of TCP in the PEP accepts out-of-order segments after the lost one, it retains them in its receive buffer to reassemble them. When the lost segment is retransmitted and received correctly, the reassembling of data is completed; the whole data consisting of the lost segment as well as the buffered segments is forwarded to the next TCP connection over the wired section. In this paper, we call this data “forward data.” Since the size of the forward data might be up to the window size of TCP over the wireless section, a large number of segments might be generated at the same time. In Fig. 1, a burst of packets is immediately sent by the PEP after the arrival of the retransmitted segment. The burst of packets should be avoided, since it may cause a buffer overflow at a node forwarding these packets on the wired section.

With regard to the busty packets from TCP, RFC 2581 specifies that TCP should employ a slow-start after a silent period of more than one RTO (Retransmission Time Out) [2]. In the normal case, when an application program issues a send request for large size data, the transmission of segments begins with the slow-start and enters the ACK clocking state after some transient period [3]. A burst of packets occurs during the period of the slow-start only. The maximum transmission rate of the burst by the slow-start is suppressed to the rate twice as large as the bandwidth of a TCP connection. However, in the case of a burst by the PEP, its transmission rate may become up to the rate of the Physical Layer, which is generally much larger than the bandwidth of a TCP connection. This means that bursts by the PEP have significant effects compared with bursts by the slow-start.

In the case of the conventional end-to-end TCP, a burst of packets of which size is equal to the TCP window may occur, if two send requests of large size data are issued within the interval of less than one RTO period. The second send request

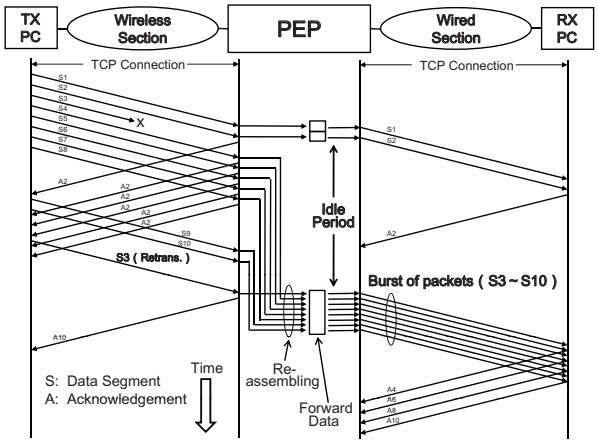


Figure 1: Data forwarding by a PEP and its output traffic.

may cause such large burst, since the slow-start is not invoked for this request. However, in the case of conventional applications based on the client-server model, a send request of large size data occurs after the previous transfer of large size data to a destination client has been completed and the client issues the next data request. As the interval between two send requests is usually larger than the RTO period, the slow-start is invoked for each send request. Therefore, the large size burst issued by TCP is not serious for the conventional end-to-end TCP.

In the case of the error recovery sequence by the PEP illustrated in Fig. 1, there is an idle period over the wired section from the loss of the segment to the reception of the retransmitted segment. However, the duration of this period is comparable with one round-trip time of the wireless section, which is much smaller than the RTO value of TCP over the wired section. Therefore the slow-start is not invoked in this case; a burst of packets derived from the large size forward data is likely fed into the wired section. Based on these reasons, it can be said that bursts of packets issued by the PEP may cause serious effects to the wired network in comparison with the case of end-to-end TCP and their mitigation is significant.

The rest of this paper is organized as follows: Section 2 discusses related work concerning PEP. Section 3 discusses the condition of burst generation and proposes a scheme to mitigate bursts of packets. Section 4 describes the simulation models and associated assumptions. Section 5 describes simulation results and discussion. Finally our conclusion is presented in Section 6.

2 RELATED WORK

Concerning the problem of the poor performance by TCP for a network including a wireless link, improvements of the congestion control done by TCP are studied. A typical version of TCP is Westwood [4], which estimates an available bandwidth. When it detects a packet loss, it identities the cause of the loss by checking whether the available bandwidth is larger than the actual transmission rate. If this is the case, the congestion control is not invoked. This approach of improving TCP has a problem that TCP in an existing PC has to be changed.

Another approach to improve the TCP performance in a wireless network is to employ local retransmissions by Link Layer protocol over the wireless section [5]. Since the Link Layer protocol hides from TCP all packet losses over the wireless section, the performance of TCP is improved. However, the modification of the protocol stack in an existing PC is needed. This approach also has the problem of spurious timeouts in TCP. Snoop [6] is a kind of local retransmission scheme, which uses duplicate TCP ACKs to invoke a retransmission of a lost packet. This scheme has an advantage of no modification of an existing PC. However, since it merely performs retransmissions of lost packets, packets arrive at the destination out-of-order. It requires a mechanism to avoid unnecessary duplicate ACKs, which is hard to implement if TCP employs SACKs.

PEPs are traditionally employed in satellite networks where a propagation delay is large. However, it is also effective to improve throughput of terrestrial radio networks of which transmission rate is increasing rapidly [7]. RFC3135 surveys various PEP architectures and their effects on the performance as well as reliability of a system. It also describes details of the controversial problem that a PEP cannot keep end-to-end semantics of TCP acknowledgement. It refers to the possibility that a PEP may send a burst of data segments due to ACK handling done by the PEP. It also describes the scheme that places a gap between ACKs to suppress the bursts. However, it never refers to the bursts of segments by the reassembling function done by the PEP.

The PEP architectures include a scheme, in which a PEP does not terminate a TCP connection, but it returns an early ACK to the sender on behalf of the destination. As the PEP does not terminate a TCP connection, bursts of segments due to the reassembling never occur in this scheme [8]. Since this scheme has to keep copies of forwarded segments in provision for segment losses for which the PEP has sent ACKs, it has problems that complex retransmission procedures as well as the buffer management are required.

Concerning bursts of packets sent by TCP for the case where TCP is employed end-to-end, bursts of packets issued by TCP are studied and classified into micro-bursts and macro-bursts [9]. However, the cause of bursts by the PEP described in the previous section is different from mechanisms described in the literature.

A natural way to mitigate the bursts of packets is to suppress the peak rate of packets at the output of TCP. In a Linux environment, a simple way is to employ TBF (Token Bucket Filter) to shape the output traffic from TCP. However, if there are multiple TCP connections, it is hard for this scheme to suppress the peak rate of each TCP connection. A scheme that suppresses the peak rate of packets precisely at the output of TCP has been proposed in Linux [10]. This scheme employs a special PAUSE frame to place a gap between two consecutive packets. However, if there are multiple TCP connections, it is hard for this scheme to suppress the peak rate of each TCP connection. In addition to this problem, although this scheme needs a large number of buffers to perform the packet pacing, it is hard to predict the required number of buffers. Packet losses due to internal buffer overflows may occur at

the pacing, if there are a large number of TCP connections.

There is a possibility that introduces the pacing function into TCP itself. Although the best performance can be expected by this approach, a modification of existing TCP implementations, which is generally hard, is needed. There is a study [11] that implements TCP as a user level library, where the rate control consistent with the TCP flow control is implemented. Although this approach has advantages that new functionalities can be easily added to TCP and development including debugging is easier than in-kernel implementation, it requires a large software development cost. A large test cost is also needed to check the conformance to the existing implementations of TCP. In this paper, we assume that the PEP is realized using existing TCP implementations in the conventional Operating Systems that have been well test and widely used.

3 THE CONDITION OF BURST GENERATION AND A MITIGATION SCHEME FOR BURSTS

3.1 The condition on bursts of packets by TCP

Since a sending side of TCP stores send data in its buffer and it performs the window flow control, a send request for large data may not directly cause a burst of packets by TCP. However, the pacing at the input of TCP can be effective for the following reasons. When TCP continues the transfer of data in a steady state, ACK clocking is effective. In this case bursts of packets are not generated, even though a send request for large data is issued. Figure 2 (a) shows the window of this condition. After TCP sends data segments of TCP window size, it waits for an ACK from the receiving side of TCP; it cannot send further segments even though a new request for send is issued. Transmission of segments is performed, only when a new ACK arrives; TCP cannot be disturbed by the arrival of a new send request. In this condition, bursts of packets never occur.

Bursts of segments can be sent by TCP, when the window is open largely as shown in Fig. 2 (b). Segments up to the open window size can be immediately send by TCP, when a send request for large data is invoked. From Fig. 1, as mentioned before, there is an idle period over the wired section from the loss of a packet to the reception of the retransmitted packet. During this period, since a number of ACKs are received by the sending side of TCP of the wired section, a substantial number of outstanding segments are acknowledged; the TCP window opens largely, before a send request for the large size forward data is issued. Therefore, a burst of segments is sent by TCP.

3.2 A proposed scheme

This paper proposes a scheme that performs the pacing at the input of TCP as shown in Fig. 3. The pacing function is performed by the application program that forwards data from TCP. This function segments the large size forward data into multiple data chunks of which size is MSS over the wired

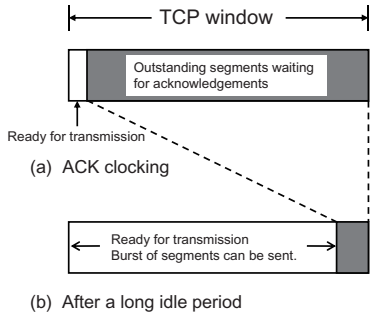


Figure 2: The condition on the occurrence of packet bursts.

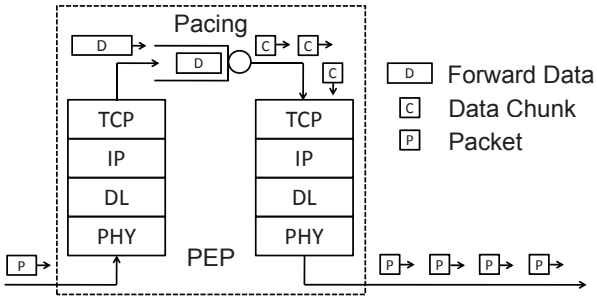


Figure 3: Relation between forward data and bursts of packets.

section. It issues a send request for each data chunk to TCP at a specific rate. This scheme has the following advantages.

- Pacing on each TCP connection can be performed independently.
- Pacing can be performed in cooperation with the flow control between two TCP connections. If large size data is forwarded from the wireless section to the wired section, the pacing function is able to stop receiving further data from the wireless section until the send requests for all chunks of the large size forward data have been completed. Because of this flow control, a packet loss due to a buffer limit never occurs at the PEP.

3.3 A pacing rate

When the pacing is employed, a problem is how to determine the suitable packet rate after the pacing. In this paper, we simply apply the packet rate derived from the transmission rate of the wireless section, assuming that the rate of the wireless section is smaller than the bottleneck rate of the wired section. There are possibilities of sophisticated schemes to determine the pacing rate, for example, estimation of end-to-end throughput over the wired section. These are left for further study.

4 SIMULATION MODELS

This paper evaluates output traffic from the PEP and effectiveness of the proposed scheme by simulations using ns-2 [12]. We compare the following three schemes.

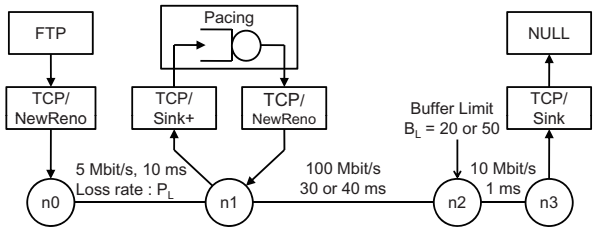


Figure 4: Simulation model for Scheme A (PEP with pacing).

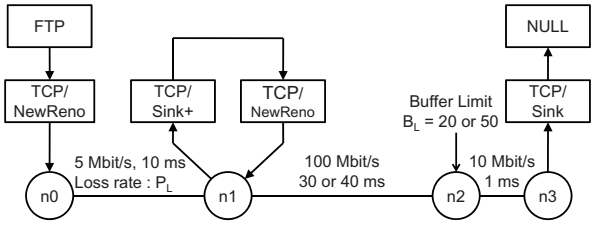


Figure 5: Simulation model for Scheme B (PEP without pacing).

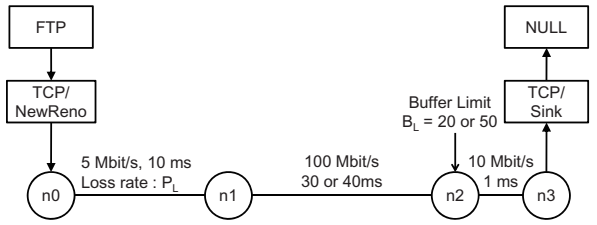


Figure 6: Simulation model for Scheme C (No PEP).

- Scheme A: PEP with pacing (proposed scheme)
- Scheme B: PEP without pacing (conventional PEP)
- Scheme C: No PEP (end-to-end TCP)

Figure 4 shows the simulation model for the proposed scheme where the PEP with pacing is employed, while Fig. 5 shows the simulation model for the conventional PEP where the pacing is not employed. NewReno TCP is employed for both the wireless and wired sections. Node n1 represents the PEP that terminates a TCP connection over the wireless section (from node n0 to node n1) and forwards received data to the next TCP connection over the wired section (from node n1 to node n3). Since original ns-2 does not have the function of delivering received data to the upper application layer, we added this function to a TCP sink in ns-2. This is indicated as "TCP Sink+" in Figs. 4 and 5. We measured the throughput of these schemes as well as the distributions of the number of packets included in a burst sent by the PEP.

The bandwidth of the wireless section is assumed to be 5 Mbit/s, while the PEP is connected to node n2 with the rate of 100 Mbit/s. Node n2 is connected to destination node n3 with the rate of 10 Mbit/s, which is assumed to be the bottleneck bandwidth B_W over the wired section. Since there is a difference of packet transmission rate between input and output in node n2, bursts of packets may be queued at this node. Moreover, we assume insufficient buffering at a node connected to the bottleneck link. The number of buffers in node n2 is limited; packets may be lost when a burst of packets is fed into this node. In order to confirm the improvement of the total throughput by the PEP we also simulate the case where a PEP is not employed. Figure 6 shows the model of scheme C (No PEP), where an error recovery is performed by TCP end-to-end. Table 1 summarizes the simulation parameters.

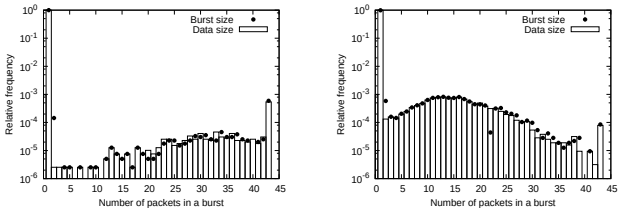
Table 1: Simulation parameters

Rate of the wireless section	5 Mbit/s
Delay of the wireless section	10 ms
TCP version for the wireless section	NewReno
TCP window for the wireless section	64 KB
Generation of packet losses	random
Packet loss rate: P_L	0.0001, 0.0002, ... , 0.05, 0.1
TCP version for the wired section	NewReno
TCP window size for the wired section	128 KB
Delay of the wired section: D_W	30 ms, 40 ms
Window size of end-to-end TCP	128 KB
The number of TCP connections	1
Rate of the wired section (PEP to n2)	100 Mbit/s
Interface rate from node n2 to n3: B_W	10 Mbit/s
The number of buffers at node n2: B_L	20, 50 packets
Simulation time	1000 sec
The number of simulation runs	10

5 SIMULATION RESULTS AND DISCUSSION

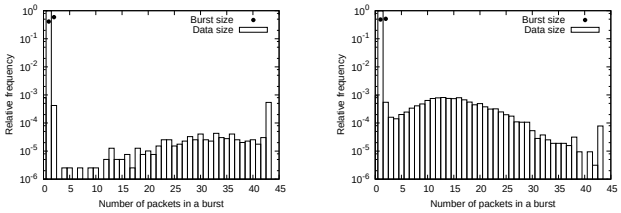
5.1 The distributions of the forward data sizes and the number of packets in a burst, where node n2 has sufficient buffering

Figure 7 shows the distributions of both the forward data sizes and the number of packets in a burst, where node n2 has enough number of buffers and the pacing is not performed. The packet loss rate P_L is selected to 0.001 and 0.01. The simulations for this figure are performed on the condition that the pacing is not performed and node n2 has enough number of buffers, which is larger than the window size of TCP over the wired section, to eliminate effects of packet losses due to buffer overflows caused by the bursts of packets. Since the maximum size of bursts can be suppressed by the window size, if node n2 has the number of buffers that is larger than the maximum window size (43) of TCP over the wireless section, a buffer overflow never occurs at node n2. The bandwidth of the bottleneck link is also twice as large as the bandwidth of the wireless section, so that the throughput of the wired section is always larger that that of the wireless section. This means that the PEP does not invoke the flow control between the wireless section and the wired section. As the original size of forward data is measured in bytes, the values of data size indicated in the figure are divided by MSS (Maximum Segment Size: 1460 B) to compare with the number of packets in a burst. In this figure rectangle boxes (Data size) represent the distribution of the forward data sizes, while



$P_L = 0.001$ $P_L = 0.01$

Figure 7: The distribution of the forward data sizes and the number of packets in a burst, where pacing is not performed.



$P_L = 0.001$ $P_L = 0.01$

Figure 8: The distribution of the forward data sizes and the number of packets in a burst, where pacing is performed.

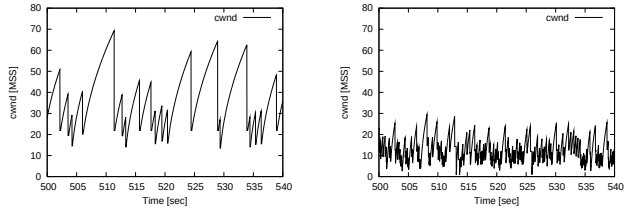
black dots (Burst size) show the distribution of the number of packets in a burst, which are sent by TCP.

The maximum size of forward data is about 43, which is consistent with MSS and the size of TCP window (64 KB) over the wireless section. We can observe that the distributions of the forward data size and the number of packets in a burst are well consistent. This means that the PEP sends bursts of packets corresponding to the forward data sizes.

In the case of the small packet loss rate (P_L = 0.001), although the frequency of burst occurrence is generally small, there is a trend that large size bursts occur compared with the case of the large packet loss rate (P_L = 0.01). The reason for this trend is as follows: When TCP detects a loss of a segment by duplicate ACKs, it retransmits the lost segment and decreases its congestion window (cwnd) in half. In the case of the large packet loss rate (P_L = 0.01), the decrease in cwnd occurs more frequently compared with the case of the small packet loss rate (P_L = 0.001). As the size of cwnd becomes small on the average and TCP window is limited by cwnd, the size of forward data in the case of large packet loss rate (P_L = 0.01) tends to be small. Figure 9 shows the time changes of the congestion window value for P_L=0.01 and P_L=0.001. In the case of P_L=0.01, the congestion window stays at around small values, where retransmissions occur frequently. On the other hand, in the case of P_L=0.001, the congestion window can have enough time to grow and take large values.

Figure 8 shows the same distributions as Fig. 7, except that the pacing is performed by the PEP. Although the distribution of the forward data sizes represented by rectangle boxes (Data size) is the same as Fig. 7, the distribution of the number of packets in a burst represented by black dots (Burst size) is completely different. The number of packets in a burst is suppressed to up to 2. This is due to the effect of the pacing. The reason of two packets in a burst after the pacing is that TCP employs the delayed ACK.

In the following subsections, parameters of the wired section, such as the number of buffers at node 2, delay of the



$P_L = 0.001$ $P_L = 0.01$

Figure 9: Examples of time changes of cwnd.

wired section and the bandwidth of the bottleneck link, will be changed. The condition of the wireless section in each subsection is always the same; the distributions of the forward data sizes for a specific packet loss rate are the same. However, the distribution of the number of packets in a burst is different depending on the conditions of the wired section. Moreover, the distribution of the number of packets in a burst is different for each packet loss rate; the distribution is not appropriate for the performance evaluation index. From this reason, we will focus on the total throughput and the packet loss rate over the wired section as the performance evaluation measures.

5.2 Throughput for the case where the number of buffers at node n2 is limited

Figure 10 shows the relation between the packet loss rate over the wireless section and the total throughput, where the number of buffers at node n2 is limited to 20. Throughput generally decreases as the packet loss rate becomes large. However, the decrease in throughput is suppressed significantly by the PEP with the pacing; scheme A achieves the best performance among three schemes. When the packet loss rate is small, the throughput of the conventional PEP (scheme B) is smaller than scheme C, in which an error recovery is performed end-to-end. The reason why the throughput of scheme B is inferior to scheme C is that packet losses occur at node n2 due to bursts of packets by the conventional PEP.

The packet losses over the wired section cause retransmissions by the fast retransmit algorithm or timeouts. The congestion widow of TCP decreases according to the fast recovery algorithm or the slow- start. The throughput of the wired section becomes small due to the decrease in the size of the congestion window. If the throughput of the wired section becomes smaller than that of the wireless section, the total throughput becomes small, since the total throughput is determined by the smaller value between the throughput of the wired section and that of the wireless section.

Figure 11 shows the relation between the packet loss rate over the wireless section and that over the wired section. Large packet loss rate is observed in scheme B, since packet losses occur due to buffer overflows at node n2.The packet loss rate over the wired section becomes larger than or comparable with that of the wireless section, when the packet loss rate of the wireless section is less than 0.005. We cannot observe any packet loss in schemes A and C. Since there is no packet loss and RTT of the wireless section becomes small due to the PEP architecture, the performance of scheme A is the best. It

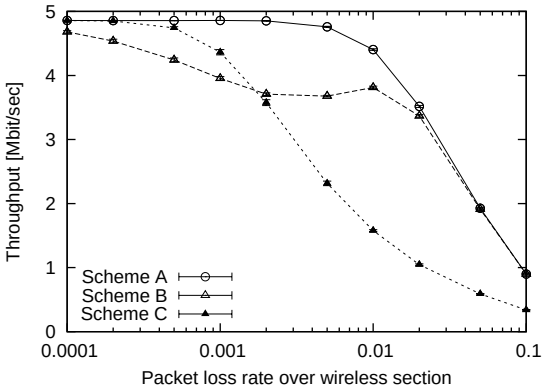


Figure 10: Packet loss rate vs. throughput, D_W = 30ms, B_L = 20, B_W = 10 M bit/s.

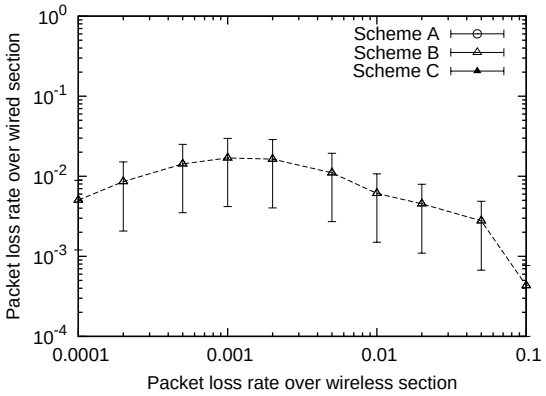


Figure 11: Packet loss rate over wireless section vs. that over wired section, D_W = 30ms, B_L = 20, B_W = 10 M bit/s.

is clear that the effect of the pacing is outstanding.

5.3 Throughput for the case where the number of buffers at node n2 becomes large

Figure 12 represents the total throughput for the case where the number of buffers at node n2 is increased from 20 to 50. Although the throughput of scheme B is much improved, its value is still lower than that of scheme A for cases of small packet loss rates (less than 0.005). When the packet loss rate is 0.005, the throughput of scheme B takes larger value than the throughput values of lower packet loss rates.

In Fig. 12, although the parameter of the wired section is changed, the throughput of scheme A is completely the same as that in Fig. 10. The reason is as follows: As mentioned before, the total throughput is determined by the smaller value between the throughput of the wired section and that of the wireless section. In scheme A, as the parameter of the wired section changes, the throughput of the wired section varies. However, its throughput is always larger than that of the wireless section. Since the total throughput is always determined by the throughput of the wireless section, the parameter of the wired section does not affect the total throughput of scheme A.

Figure 13 shows the relation between packet loss rate of

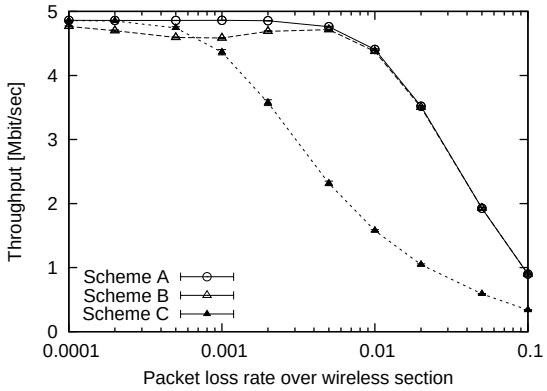


Figure 12: Packet loss rate vs. throughput, D_W = 30ms, B_L = 50, B_W = 10 M bit/s.

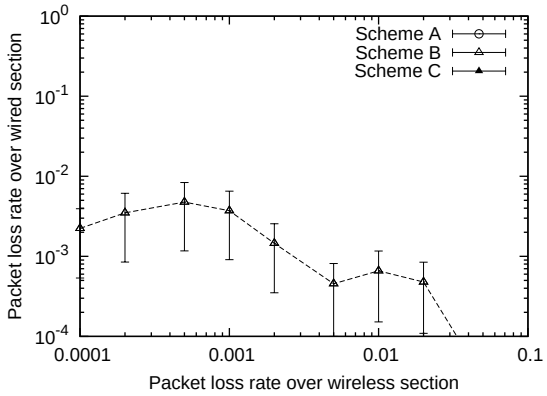


Figure 13: Packet loss rate over wireless section vs. that over wired section, D_W = 30ms, B_L = 50, B_W = 10 M bit/s.

the wireless section and that of the wired section. Due to the increase in the number of buffers, the packet loss rates of scheme B become smaller than those of Fig. 11. Although the packet loss rate over the wired section becomes small in comparison with the case of Fig. 11, its value is still larger than that of the wireless section for the cases of the packet loss rates less than or equal to 0.001. Similar to the case of Fig. 10, this explains the degradation of the throughput when the packet loss rate of the wireless section is small. As there is no packet loss in schemes A and C, the performance of these schemes is the same as the case of Fig. 10.

5.4 Throughput for the case where the delay of wired section becomes large

Figure 14 shows the relation between the packet loss rate over the wireless section and the total throughput, where the delay of the wired section is 40ms, which is 10ms larger than the case of Fig. 10. The throughput of scheme B becomes worse while the throughput of scheme A is the same as the case of Fig. 10. In the case of scheme A, although the maximum throughput of the wired section becomes small due to the increase in the delay of the wired section, its value is still larger than the throughput of the wireless section. Then, the total throughput scheme A is the same as the case of Fig. 10. Since the increase in the delay of the wired section is relatively small compared with with the total end-to-end delay,

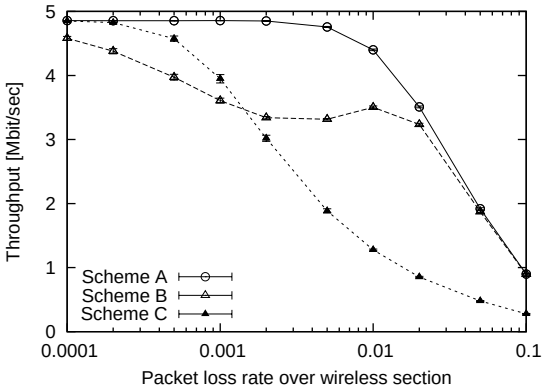


Figure 14: Packet loss rate vs. throughput, D_W = 40ms, B_L = 20, B_W = 10 M bit/s.

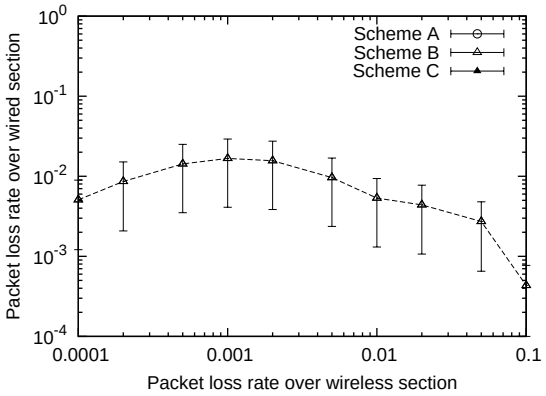


Figure 15: Packet loss rate over wireless section vs. that over wired section, D_W = 40ms, B_L = 20, B_W = 10 M bit/s.

the throughput of scheme C is also almost the same as Fig. 10.

In Fig. 14, although the parameter of the wired section is changed, the throughput of scheme A is completely the same as that in Fig. 12. The reason is as follows: The total throughput is determined by the smaller value between the throughput of the wired section and that of the wireless section. In scheme A, as the parameter of the wired section changes, the throughput of the wired section varies. However, its throughput is always larger than that of the wireless section. Since the total throughput is always determined by the throughput of the wireless section, the parameter of the wired section does not affect the total throughput of scheme A.

Figure 15 shows the relation between the packet loss rate over the wireless section and that over the wired section. This figure is almost the same as Fig. 11. As the RTT of the wired section increases, the effect of packet losses to the throughput becomes large. The reason is that losses of packets cause decrease in the congestion window (cwnd), which leads to the decrease in throughput. This indicates that the effect of bursts of packets by the conventional PEP becomes serious when the delay of the wired section becomes large.

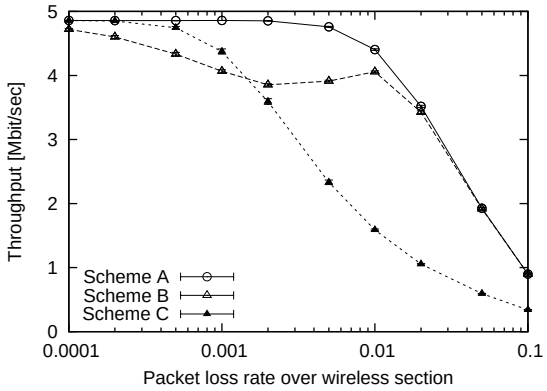


Figure 16: Packet loss rate vs. throughput, D_W = 30ms, B_L = 20, B_W = 20 M bit/s.

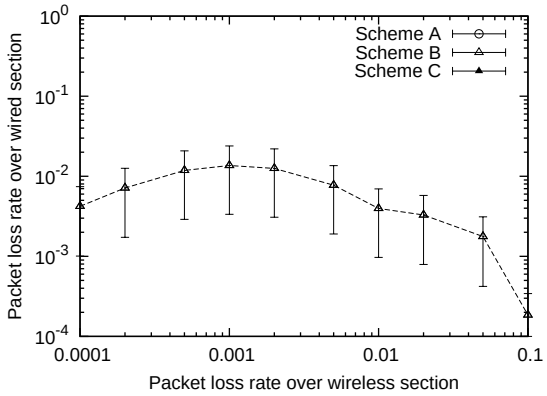


Figure 17: Packet loss rate over wireless section vs. that over wired section, D_W = 30ms, B_L = 20, B_W = 20 M bit/s.

5.5 Throughput for the case where the bandwidth of the bottleneck link becomes large

All the results presented above are taken on the condition that the bandwidth of the bottleneck link B_W is 10 Mbit/s. Figure 16 shows the relation between the total throughput and the packet loss rate over the wireless section, when the bandwidth B_W is doubled (20 Mbit/s) and the number of buffers at node n2 is 20. The throughput of scheme A and C is the same as the case of Fig. 10, since there is no packet loss in Fig. 16 and Fig. 10. We can observe improvement of throughput of scheme B compared with Fig. 10. However, the throughput of scheme B is still the smallest among the three schemes for the cases of small packet loss rates.

Figure 17 shows the relation between the packet loss rate of the wireless section and that of the wired section. Although the packet loss rate of the wired section is improved, its value is still large. We cannot observe significant effectiveness of the increase in the bottleneck bandwidth. This shows that the increase in the bandwidth of the bottleneck link has small impact on suppressing packet losses due to a large burst of packets.

6 CONCLUSION

This paper has investigated the architecture of a TCP proxy (PEP), where the PEP forwards receive data from one TCP connection on the wireless section to another TCP connection on the wired section. We have focused on the output traffic from the PEP. When losses of packets occur due to transmission errors, we have identified a problem that the output traffic from the PEP becomes bursty due to the reassembling function done by the receiving side of TCP over the wireless section. To mitigate the bursts of packets we proposed a pacing function at the PEP. This scheme has advantages of no packet loss with a limited amount of buffers and independent pacing for each TCP connection. We simulated the case where the output traffic from the PEP is forwarded by a node that is connected to a bottleneck link in the wired section and the node has insufficient number of buffers compared with the window size of TCP. The simulation results have shown that the performance of the PEP with pacing is improved significantly. We have also observed that the total throughput of the conventional PEP becomes worse than that of end-to-end TCP for the cases of small packet loss rates. As the rate of wireless links is increasing, a TCP proxy will be important to attain high throughput. Accordingly, a burst of packets by the TCP proxy will be serious and its mitigation will be significant.

ACKNOWLEDGMENT

This work has been supported by KAKENHI (20500079) and Special Research Grant in Aids of Fukui University of Technology.

REFERENCES

[1] J. Border, M. Kojo, J. Griner, G. Montenegro, and Z. Shelby, “Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations,” IETF, RFC3135, June 2001.
[2] M. Allman, V. Paxson, and W. Tevens, “TCP Congestion Control,” RFC 2581, April 1999.
[3] W. R. Stevens, “TCP/IP Illustrated, Volume 1: The Protocols,” Addison-Wesley, 1994.
[4] Casetti, M. Gerla, S. Mascolo, M. Y. Sanadidi, and R. Wang, “TCP westwood: Bandwidth estimation for enhanced transport over wireless links,” ACM MOBICOM 2001, pp. 287–297, July 2001.
[5] G. Fairhurst and L. Wood, “Advice to Link Designers on Link Automatic Repeat reQuest (ARQ),” IETF, RFC3366, August 2002.
[6] H. Balakrishnan, S. Seshan, E. Amir, and R. Katz, “Improving TCP Performance over Wireless Networks,” 1st ACM International Conference on Mobile Computing and Networking (Mobicom), Nov. 1995.
[7] M. Meyer, J. Sachs, and M. Holzke, “Performance evaluation of a TCP proxy in WCDMA networks,” IEEE Wireless Commun., vol. 10, no. 5, pp. 70–79, Oct. 2003.
[8] D. Dutta and Y. Zhang, “An Active Proxy Based Architecture for TCP in Heterogeneous Variable Bandwidth Network,” IEEE GLOBECOM 2001, pp. 2316–2320, Nov. 2001.
[9] M. Allman and E. Blanton, “Notes on Burst Mitigation for Transport Protocols,” ACM Computer Communications Review, vol. 35, no. 2, pp. 53–60, April 2005.
[10] R. Takano, T. Kudo, Y. Kodama, M. Matsuda, F. Okazaki, and Y. Ishikawa, “Improving TCP Performance by Using Precise Software Pacing Method,” IEICE Technical Report, NS2005-157(2006-1).
[11] Y. Miyake, T. Kato, and K. Suzuki, “Implementation Method of High Speed Protocol as Transport Library,” ICNP, pp. 172–179, Nov. 1995.
[12] The Network Simulator – ns-2, http://www.isi.edu/nsnam/ns/, 2010.

(Received July 6, 2010)
(Revised February 29, 2012)

Toshihiro Shikama received his B.E. and M.E. degrees from Tokyo Institute of Technology in 1974 and 1976, respectively. From 1984 to 1985, he was at University of Waterloo. He received his Ph.D. degree from Shizuoka University in 2006. He joined Mitsubishi Electric Corp. in 1976 and had engaged in developing a computer network using a satellite channel, high speed ring type LANs, time division multiplexers, ATM equipment, a high speed IP switch, and network security systems. Since April of 2007, he has been a professor at Department of Electrical, Electronic and Computer Engineering, Fukui University of Technology. He is a member of IPSJ, IEICE, and IEEE Communications Society.