A Reliable TCP-Aware Link Layer Retransmission for Wireless Networks

Zhu Jing Niu Zhisheng
State Key Lab on Microwave and Digital Communication
Dept. of Electronic Engineering TSINGHUA UNIVERSITY Beijing 100084 China
Email: ziju@atm.mdc.tsinghua.edu.cn
niuzhs@mail.tsinghua.edu.cn

Abstract: In this paper, we develop a new TCP-aware link layer retransmission scheme that can offer almost full reliable transmission over wireless channels and efficiently avoid the adverse interaction of different layer protocols. We introduce time-stamp in our scheme to offer reliable delivery for out-of-order segment and use explicit retransmission notification (ERN) to avoid false window deflating and multiple retransmission of TCP source. By simulation, we conclude that our proposal can greatly improve TCP performance over wireless networks.

Key Words: Wireless Network, Internet

1. Introduction

With the fast development of Internet, it becomes more and more important to access Internet through mobile communication systems. ITU are now making great efforts to promote international standards of the third generation mobile communication system (IMT2000), which is expected to support mobile computing and nomadic access of Internet.

TCP is originally designed for wireline networks and has been widely used in today's Internet. But it is facing many challenges in wireless networks. Therefore, many proposals are provided to give some improvements, such as TCP-NewReno [3] and SACK [4]. But all of them are end-to-end solutions and have some unresolved problems such as slow reaction to packet loss due to long round trip time and false window adjustment due to regarding all packet loss as the result of congestion.

To solve these problems, some local solutions are provided such as AIRMAIL [5] and other ARQ/FEC based link layer protocols. They improve TCP performance by retransmit lost segments locally. While these pure local solutions need a large mount of additive load for reliable transmission and therefore decrease efficiency. To solve this problem, the idea of TCP-awareness is introduced in Snoop [2], which is to offer local retransmission by using TCP acknowledgment packet at link layer. But it still uses a timer to recover out-of-order packet losses, and this requires accurate estimation of round trip time of a wireless link. Furthermore, setting timer in a base station wastes large system resources. In addition, the way of suppressing duplicated ACK can not completely avoid unnecessary timeout of TCP source.

Either congestion or retransmission loss may lead to out-of-order delivery. If a out-of-order packet is lost in a wireless link, it can not be indicated by using duplicated ACK or partial ACK [3]. Thus, one of the popular solutions is using timeout retransmission [2]. However, this requires accurate estimation of round trip time of a wireless link that is difficult in an unreliable network.

Another way to this problem is to use a pure link layer protocol such as ARQ, which can give each packet a new sequence number to identify arrival sequence of packets and uses it to offer loss recovery. But the problem of overhead may result in a low efficiency.

In this paper, we focus on how to efficiently recover from out-of-order packet loss and avoid false actions of TCP source. To solve these two problems, we introduce cooperation of the transport layer and the link layer. That is making link layer transmission sequence work together with TCP segment sequence to detect loss, then using explicit retransmission notification to avoid false actions of TCP source. Also,
we base our scheme on TCP-aware local retransmission.

Our paper is organized as follows. In Section 2, the details of our scheme are showed. In Section 3, the simulation results are given. We conclude the whole paper in Section 4, and direct the future work.

2. Reliable TCP-Aware Retransmission at Link Layer

The key idea of our scheme is taking advantages of TCP and link layer coupling in order to reduce link layer overhead and offer efficient reliable delivery of out-of-order packet. In our scheme we use time-stamp, which can be looked as a link layer sequence number of TCP segment, to indicate transmission sequence of TCP segment in a wireless access point.

In a wireless access point, each TCP segment is encapsulated into one LAC-PDU (Link Access Control – Protocol Data Units). If a non-transparent RLC mode of a IMT-2000 system [6] is used, each LAC-PDU has its own overhead (say 3 octets) typically consisting of at least a service access points identifier and sequence number for higher-lever ARQ and other fields. A time-stamp is inserted in the LAC overhead. Combining it with ACK packets of TCP, our loss detection algorithm can detect out-of-order packet loss on a wireless channel. This is why our scheme does not need timer.

The detail of our scheme is as follows. On reception of a TCP segment from a fixed terminal to a mobile host at a base station, our scheme will encapsulate the TCP segment into a LAC-PDU and insert a time-stamp into the LAC overhead, then save its copy in the buffer and send it to the mobile host. When a mobile host successfully receive that TCP segment, it will send back a TCP ACK packet. Each TCP ACK packet is also encapsulated into a LAC-PDU, and the time-stamp of the received TCP segment will be inserted into overhead of TCP ACK packet’s LAC-PDU by the mobile host. Using the time-stamp feed-backed by a TCP ACK packet, we can know whether the packet transmitted after a special out-of-order packet has been received successfully. According to this, we can quickly detect lost packet although it may be an out-of-order packet and therefore need not wait for timeout. This way is something like what TCP SACK does for loss recovery. On the other hand, we use partial ACK and duplicated ACK to detect in-order packet loss. In addition, when a TCP segment is retransmitted by our scheme at link layer, it will be added a new time-stamp to indicate current transmission sequence and moved from current place to the tail of the buffer.

Window deflating is a method used by TCP to avoid congestion. But because TCP can't classify the packet lost in a congestion node with other lost in a wireless link, those packets lost in a wireless link will trigger false window deflating. This will make the congestion window size of TCP maintain a low value and throughput degrades greatly. In addition unnecessary retransmission of TCP source will waste network resources. In order to avoid false actions of TCP source we use duplicated ACK packets as carriers of ERN (Explicit Retransmission Notification) bit to prevent from possible multiple retransmissions and false window deflating of TCP source. If the packet expected by duplicated ACK received at a base station is in the buffer, we will mark ERN in this duplicated ACK. If TCP source has received duplicated ACK with ERN and then fast retransmission or timeout is triggered, it can maintain window size and cancel unnecessary retransmission. Much research has showed that ELN [7] (Explicit
Loss Notification) bit of TCP ACK packet can be used to identify packets lost in a wireless channel from those lost in a congestion node. Here ELN also can be used as ERN in our scheme. In addition, we also can use one of six reserved bits in a TCP packet head as ERN but this way still needs to be further negotiated.

3. Simulation and Analysis

For simulation of TCP performance we use ns2. Fig. 1 shows our network scenario for simulation. We use random loss model in a wireless link. For simplification, we only consider a typical single TCP connection. In addition, Data are transmitted from a fixed TCP source to a mobile host. In our simulation, we use RLP (Reliable Link Protocol) to denote our scheme. Some TCP source variables are set as follows:

TCP data packet size = 1000 bytes
TCP ACK packet size = 40 bytes
Maximum congestion window = 40 packets

1) Improvement on Throughput of End-To-End TCP Connection

To show the loss recovery ability of our new proposal, the buffer size of a congestion node in our simulation is 100 packet that is larger than maximum congestion window size of TCP source. Therefore, there will be no congestion loss and all lost packets is due to wireless environment.

In Fig. 2, we compare our proposal with other end-to-end schemes in different PER (Packet Error Rate). We define the total number of effective segments received by mobile host in the whole simulation period as throughput. The result shows that the performance of end-to-end scheme degrades greatly when PER is greater than 0.01, but our new proposal can work well until PER is 0.1. This shows that our scheme can prevent from TCP performance degradation in a wireless environment.

2) False Actions of TCP Source

In this section we consider the effect of false window deflating and multiple retransmission on TCP performance.

In simulation, the buffer size of router is 20 packet that is smaller than maximum congestion window size. Therefore some packets may be lost. In addition we use TCP Reno as the end-to-end scheme that is cooperated with our TCP-aware link layer retransmission scheme in simulation. Fig. 3 shows the trace of window changing of TCP when using different proposals. From this figure we can see that because of false window deflating TCP will maintain a small window size. While using explicit retransmission notification can avoid false window deflating. Fig. 4 shows throughput comparison when PER is gradually increased. We can see that the explicit retransmission notification can improve performance.

![Fig.2 Throughput against Packet Error Rate (Simulation Time = 15s)](image-url)
Multiple retransmission is mainly due to fast retransmission. It wastes network resources and impacts the end-to-end performance of TCP. But through our simulation, we find that the effect of multiple retransmission on throughput is little. Fig.5 shows the simulation results. Therefore, we can ignore the effect of multiple retransmission.

3) Full Reliable Scheme vs. Partial Reliable Scheme

In some environment time-stamp probably can not be supported and out-of-order packet can only be recovered by TCP source. This means that our scheme only can offer partial reliable delivery. Through simulation, we find congestion loss (Fig.6b) and high PER (Fig.6a) make the scheme without timestamp work worse. But because using TCP-aware local retransmission, it still can acquire the more satisfactory performance than all end-to-end solutions.

In addition, Fig.6b shows that the throughput (PER 0.1) is greater than throughput (PER 0.05 and PER 0.01). This indicates that wireless loss can avoid congestion loss. We can explain it as follows. Although our new proposal can quickly recover lost packets, new TCP acknowledgment is still delayed and this will decrease the speed of packet transmission and avoid overflowing of the buffer in the congestion node. This is related to interaction of congestion loss and wireless loss, and we will not talk about it in this paper.
Fig. 5 Throughput against PER (Buffer Size of Router = 20 packets)

Fig. 6a Throughput against PER (Buffer Size of Router = 100 packets)

Fig. 6b Throughput against PER (Buffer Size of Router = 20 packets)
4. Conclusion

Through our research, we propose a new TCP-aware local retransmission scheme. By introducing time-stamp at link layer, we can offer full reliable delivery in a wireless link for out-of-order packet, and by introducing ERN (Explicit Retransmission Notification) we avoid false actions of TCP source. In future work, we will build up a more accurate simulation environment for the next generation mobile system. Some MAC protocol will be considered, and an error-model of a wireless channel in W-CDMA will be designed. Finally, we expect to give a complete proposal for improving TCP over the future mobile system.

Reference: