LT-TCP: End-to-End Framework to Improve TCP Performance over Networks with Lossy Channels

Omesh Tickoo[1], Vijaynarayanan Subramanian[1], Shivkumar Kalyanaraman[1] and K. K. Ramakrishnan[2*]

[1] Dept. of ECSE - RPI, {tickoo, subrav, kalyas}@rpi.edu
[2] AT&T Labs Research, kkrama@research.att.com

Abstract. TCP performance over wireless links suffers substantially when packet error rates increase beyond about 1% - 5%. This paper proposes end-end mechanisms to improve TCP performance over lossy networks with potentially much higher packet loss rates. Our proposed scheme separates congestion indications from wireless packet erasures by exploiting ECN. Timeout effects due to packet erasures are combated using a dynamic and adaptive Forward Error Correction (FEC) scheme that includes adaptation of TCP's Maximum Segment Size. Proactive and reactive FEC overhead enhance TCP SACK to protect original segments and retransmissions respectively. Dynamically changing the MSS tailors the number of segments in the window for optimal performance. SACK and timeout mechanisms are used as a last resort. *ns-2* simulations show that our scheme substantially improves TCP performance even for packet loss rates up to 30%, thus extending the dynamic range and performance of TCP over networks with lossy (e.g., wireless) links.

1 Introduction

With the use of WiFi (802.11) hotspot/metro access, WiMax (802.16), 3G, mesh and community wireless networks, end-to-end communication could involve traversal of multiple wireless links. In such links, performance variability is the norm: TCP will see variable capacity and unpredictable *residual* packet erasure rates. Seamless communication under such conditions requires tolerance of such performance variability, especially packet erasures.

TCP depends on packet loss to respond to congestion, and its drawbacks over lossy wireless links are well-known. A key issue is TCP's inability to distinguish between losses due to channel errors and congestion, leading to significant underestimation of the available capacity. This behavior only worsens as the channel error rate increases. It is important to separate TCP's response to congestion from packet erasures.

* This work was supported in part by grants from AT&T Labs Research, Intel Corp., NSF (grant number NSF-ITR 0313095) and ARO (grant number W911NF-04-1-0300).

Explicit Congestion Notification (ECN) is a mechanism that can be used to unambiguously indicate incipient congestion. By sharply reducing congestion loss (due to buffer overflow), it allows us to isolate packet losses as being due, primarily, to channel errors. In this paper, we re-examine TCP's behavior in ECN-enabled networks and propose adaptive mechanisms that allow robust performance even under heavy and persistent erasure conditions (e.g., up to 30% erasure rates). With TCP reacting to ECN [10], packet loss in a network with wireless links would be predominantly due to bit errors. However, the resultant packet erasures stills extract a substantial performance toll through TCP timeouts. We therefore propose a package of complementary and adaptive mechanisms (adaptive MSS and proactive/reactive FEC) to recoup TCP's performance, with minimal end-end extensions.

An interesting question is: Why end-to-end mechanisms for erasure tolerance over-and-above link-level error protection mechanisms? First, link level mechanisms may not be sufficient. Recently, studies by a group of researchers showed substantial *residual* performance variability (e.g., 10-50% packet erasure rates) in 802.11b mesh networks [1]. Emerging high speed LAN standards like 802.11n use adaptive modulation/coding techniques (i.e., variable capacity) targeting a packet error rate of 10%, but these techniques are triggered by low SNR events (i.e., bursty packet erasures). The efficacy of ARQ persistence in 802.11x is countered by the exponential backoff timers, leading to variable capacity/delays. Barakat et al [7, 8] study TCP over links with just FEC or hybrid ARQ/FEC. They find a pure FEC strategy ineffective. Pure ARQ is also shown to fail for high erasure conditions, despite persistent retries. Though link-level hybrid ARQ/FEC is better than either FEC or ARQ alone, its performance also significantly degrades for higher loss rates (5% or more) despite high amounts of ARQ retries, fragmentation of IP packets, FEC overhead and buffering (see Fig. 15/16 in [8]). The situation is complicated further because different link layer standards/implementations have different erasure resilience capabilities.

Second, any appreciable residual erasures may have a disproportionate impact on TCP depending upon *which* packets are lost (e.g., data, acks, or retransmissions). Erasures of retransmissions or segments when TCP's window is small raise the risk of timeouts. In addition, information about the current window size, loss rate and packet size (MSS) can be exploited by TCP to provide the correct and variable amount of error protection when needed. Of course, our design (or the end-end design principle) does not preclude *general-purpose* error mitigation schemes at the link layer, and we remain cautiously optimistic about the potential of link-layer hybrid ARQ.

TCP Performance Enhancing Proxies (PEPs) [6] are TCP-aware mechanisms placed on boundaries where network characteristics change dramatically. PEPs maintain per-flow state and perform layer violations (with implications for security, mobility and scalability). The TCP-PEP technique is less applicable for the emerging regime of variable-performance, high-erasure, highly multiplexed, meshed wireless links.

Rizzo showed the feasibility of transport-layer high-speed FEC computation[5]. Although [5] mentions the idea of FEC in TCP, a specific scheme has not been studied and subsequent researchers' focus has been on multicast transport protocols [2, 4]. Recent attempts at FEC with TCP have met with limited success [3] (for less than 10% erasure rates). Success with higher erasure rates have not been reported to the best of our knowledge. TCP Westwood [9] uses an estimate of output rate to guide congestion control, and has been effective for low erasure rates (under 5%). Presumably all these schemes encounter the risk of increased timeouts mentioned earlier. Overall, despite growing interest, there has been no clear baseline proposal that offers a significant increase in TCP performance over a wide range of erasure rates.

In our scheme, called **Loss-Tolerant TCP (LT-TCP)**, we provision *proactive* FEC in the original window as a function of the estimate of the actual packet erasure rate (PER). *Reactive* FEC is used to mitigate the effect of erasures during the retransmission phase. An adaptive maximum segment size (MSS) component provides a minimum granularity (a minimum number of packets) in the TCP window, again seeking to reduce the risk of timeouts. We seek to adaptively balance the FEC and packetization overhead while reducing the risk of timeouts and also rapidly recovering erased packets. In particular, when the end-to-end path has little or no loss/erasure, LT-TCP introduces negligible overhead. At the same time, we seek to significantly improve the performance of TCP and channel utilization even under packet erasure rates as high as 30-50 percent.

The rest of this paper is organized as follows. Section 2 describes the scheme. Performance results (ns-2 simulations) are presented in Section 3. The last section presents our conclusions and future work.

2 Scheme Description

LT-TCP design focuses on the following key issues:

- **Congestion Response:** How should TCP respond to congestion, but *not* respond to packet erasures. What is the appropriate signal of congestion in an error-prone environment?
- **Mix of Reliability Mechanisms:** What mix of TCP repair mechanisms (ARQ, FEC) should be used to achieve the TCP reliability objectives and how should they be structured?
- **Timeout avoidance:** Timeouts are a final fallback mechanism under significant congestion loss, but truly wasteful otherwise. How can the mix of TCP repair mechanisms be setup to reduce the timeout risk ?

Congestion Response: Our answer to this issue is simple: *react only based upon ECNs, not on detection of packet loss.* This solution would obviously work only in an ECN-enabled network. However, despite this simplifying assumption, timeout risk reduction poses further challenges as discussed below.

Reliability Mix: Error correction packets (a.k.a. FEC packets) have a property unlike regular data packets: if *any* k (out of N) packets are received, then

it does not matter *which* k packets are received. A *unique* FEC packet can repair any one data packet. In contrast, TCP uses SACK or 3-dupacks to identify and retransmit a packet with a *specific* sequence number. This sequence-agnostic property for FEC-based repair allows a unique FEC packet to be used either in the original window (i.e., in a proactive manner, called **PHASE 1**) or in the retransmission process (i.e., in a reactive manner, called **PHASE 2**). If the *cumulative number* of FEC and data packets in PHASE 1 and PHASE 2 do not meet the threshold of k (out of N), we will fallback to traditional retransmission or timeout. Our mix will first have adaptive amounts of proactive and reactive FEC repair packets, extending the traditional TCP mechanisms (SACK, dupacks, timeouts, retransmissions).

Timeout avoidance: Timeouts occur for the following key reasons that are exacerbated in a high packet erasure environment:
a) *All* packets in a window are lost.
b) *Three dupacks* do not reach the source (to trigger SACK-based repair).
c) One or more of the retransmitted packets are lost (because dupacks stop arriving).
To overcome each of these issues related to timeout avoidance, we propose to:
a) Granulate the TCP window more finely to increase the number of segments in a window that (due to the self-clocking nature of TCP) are spread over an RTT. Smaller packets also reduce the impact of bit errors (which translate to smaller packet error rates).
b) Use proactive FEC packets in the window based upon an estimate of current erasure rate to reduce the need for dupacks and reduce the burden on SACK retransmissions for recovering lost packets.
c) Use reactive FEC repair packets triggered by dupacks to complement and protect SACK retransmissions.

In summary, we propose the following complementary building blocks to extend TCP-SACK:

ECN-Only: Congestion response only to ECN, since it is the definitive signal of congestion in ECN-enabled networks.

Per-Window Erasure Rate Estimate (E) We use an exponential weighted moving average (EWMA), with adaptive parameters (to increase responsiveness and bias towards higher estimates after a spike):

$$E = \alpha \times new_l + \beta \times E \tag{1}$$

$$\alpha = \frac{new_l}{new_l + E}, \quad \beta = 1 - \alpha = \frac{E}{new_l + E} \tag{2}$$

Section 3 shows that Equation 1 tracks the average erasure rate fairly well, although it may overestimate the erasure rate after a spike (burst loss). Under such conditions the proactive FEC algorithm will add deadweight FEC that is either insufficient to provide required protection or is more than the level required. Since previous studies have noted that the erasure rates

are relatively stable over intervals as large as a second [1], we feel that the estimate we use will track the actual erasure rate fairly closely over most wireless channels. The erasure rate estimation can be performed equally conveniently at either the receiver or the sender. The receiver can use the information from the packets received to estimate E while the sender can use the ACK information to do the same.

Proactive FEC: The number of FEC packets per window (P) used in PHASE 1 (i.e., Proactive FEC) is a function of the erasure estimate, i.e., $P = f(E)$ The MSS is adjusted to allow one or more FEC packets per window (while maintaining sufficient window granulation). Our initial method divides the erasure rate range into multiple *bins*. Depending on the bin E falls in, we select a hard-coded number of FEC packets that define the minimum number of proactive FEC packets needed. We are investigating alternate methods to better decouple the amount of FEC added from the granulation decision.

Adaptive MSS: Granulate the congestion window to have at least G packets, subject to limits of a minimum and maximum MSS (MSS_{min} and MSS_{max}). Depending on the window size in bytes, the MSS is adjusted to accommodate the required number of FEC packets while providing adequate erasure protection. Thus, the variation in MSS is governed by the following factors:

- The window must be large enough to maintain the minimum granularity, G.
- The window should be able to accommodate at least $f(E)$ proactive FEC packets while providing adequate erasure protection for the estimated erasure rate, E.
- The MSS chosen must be bounded by the MSS_{min} and MSS_{max} values.

Reactive FEC: For every dupack, the sender transmits R reactive FEC packets. R is a function of the erasure rate estimate, E. i.e., $R = g(E)$. Again, R is currently chosen depending on the erasure *bin* that E falls in. The reactive FEC packets will complement and protect data in PHASE 1 and SACK retransmissions in PHASE 2.

The sender module is responsible for adaptive MSS adjustment (i.e. window granulation), computing proactive and reactive FEC packets, and the appropriate transmission of FEC packets.

The receiver implements packet reconstruction (using FEC if and when necessary) and per-window loss-rate estimation. The FEC overhead (proactive and reactive) is computed on a per-window basis using shortened Reed-Solomon (R-S) codes (similar to the method used in CD-ROMs). The proactive FEC is transmitted in the window, but the inventory of excess FEC packets is stored for potential use as reactive FEC.

The tradeoffs of our mechanisms are as follows. Adaptive MSS uses smaller segments when windows are small and therefore the header (or packetization) overhead is larger then, but diminishes as window sizes grow. Proactive FEC may lead to a small deadweight goodput degradation due to over-estimation of erasure rate, and some increased burstiness in the release of dupacks from the destination. Reactive FEC triggered by each dupack leads to a somewhat

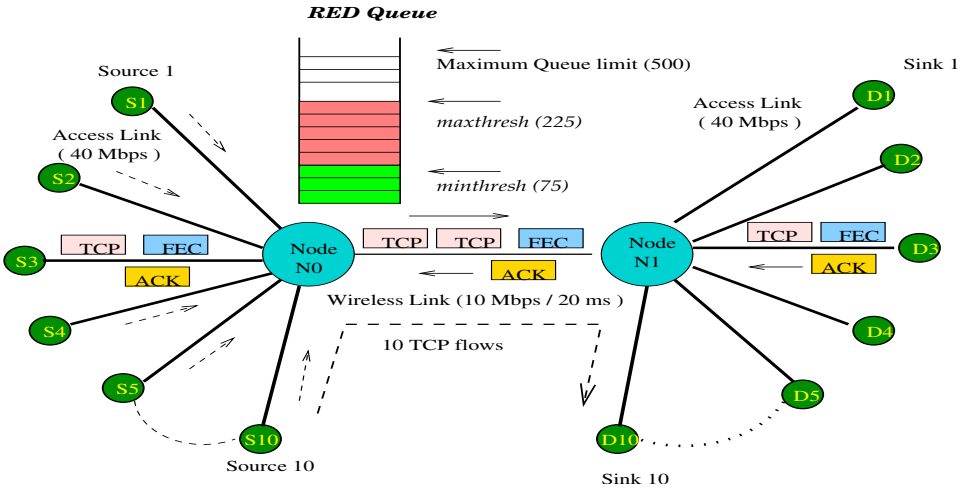


Fig. 1. Single Wireless Bottleneck Setup: RED AQM with ECN.

increased load and burstiness in the retransmission periods. However, since these mechanisms are all adaptive (i.e., they become more active only during higher erasure rate conditions), we argue that the tradeoffs are worth making as they achieve a significant improvement in performance, and enables a wider dynamic range of applicability of TCP.

3 Performance Results

In this section, we present the performance of LT-TCP compared with TCP-SACK (with ECN), the performance of LT-TCP components, comparisons of LT-TCP and two link level schemes (LL FEC and LL Hybrid FEC-ARQ). The link layer schemes are as follows: LL FEC uses FEC to match the average packet erasure rate (PER) on the link and LL Hybrid FEC-ARQ is a hybrid ARQ/FEC scheme that uses 10% FEC protection and has ARQ persistence of 3. LT-TCP performs better than all the schemes compared, especially as the PER increases (up to 30-40%).

We use a single-bottleneck test case (see Fig. 1: 10 Mbps bottleneck, 20 ms one-way delay, 10 TCP flows) with erasure rates varying from 0% to 50% is used. Hosts are ECN-enabled, bottlenecks implement RED/ECN on a 250 KB buffer (i.e. upto 500 packet of size 500-bytes). RED *minthresh* and *maxthresh* values are 75 and 225 packets respectively. The simulations are run for 1000 seconds, and results are averaged over 5 randomized runs. To assess the contribution of LT-TCP components, we use a 10% PER test case.

Metrics include aggregate throughput, goodput, congestion window dynamics, number of timeouts, bottleneck queue dynamics, FEC overhead and adaptive MSS dynamics.

3.1 LT-TCP vs TCP-SACK

Tables 1 and 2 present the performance of TCP-SACK and LT-TCP respectively. TCP-SACK and LT-TCP perform well *without* packet erasures. But TCP-SACK's performance drops quickly for PER of 10% and higher. LT-TCP outperforms TCP-SACK by a wide margin and its absolute performance (goodput) is good up to about 30% PER (see Table 2). However, for higher PER (40% and higher) the goodput drops off, while the number of timeouts goes up, despite high FEC overhead. This points to room for further improvements to LT-TCP.

In comparison however, TCP-SACK is worse. The congestion window dynamics shown in Fig. 2(b) shows that at 20% PER, TCP-SACK is operating with a very small window compared to LT-TCP. TCP-SACK sees fewer total number of timeouts at high erasure rates. But this is due to Karn's exponential timer back-off algorithms (triggered with back-to-back timeouts), and it spends significantly more time in each timeout period, achieving very little useful goodput.

	ERROR RATE					
PARAMETER	0 %	10 %	20 %	30 %	40 %	50 %
Goodput(Mbps)	9.158	1.098	0.233	0.048	0.01	0.003807
Number of Timeouts	0	267	287	135	52	26
Throughput (Mbps)	9.52	1.272	0.306	0.073	0.018	0.007984

Table 1. TCP-SACK w/ Erasure Rates (0-50%)

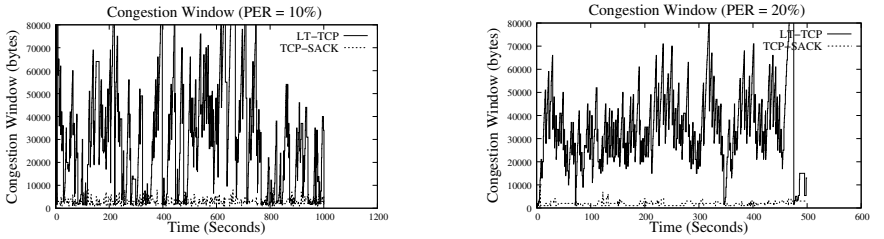
	ERROR RATE					
PARAMETER	0 %	10 %	20 %	30 %	40 %	50 %
Goodput(Mbps)	8.94	5.36	4.086	2.99	0.89	0.3
Number of Timeouts	1	24	19	40	130	243
Throughput(Mbps)	9.53	8.55	9.01	9.06	3.53	1.74
Proactive FEC Overhead (%)	2	29	45	52	53	55
Reactive FEC Overhead (%)	0	3.7	7	11	15	17

Table 2. LT-TCP w/ Erasure Rates (0-50%)

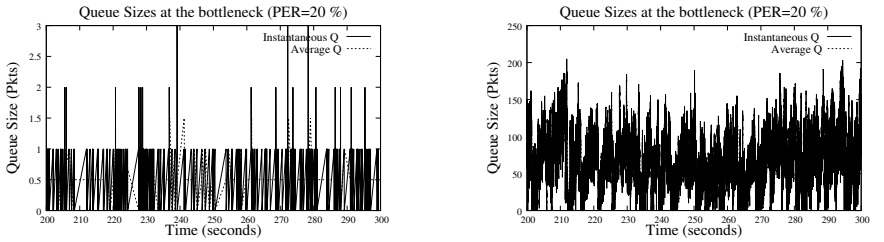
The congestion window (cwnd) and queue graphs reiterate these points. With TCP SACK at high PER (10-20%), cwnd is small and queues are small (i.e. bottlenecks are underutilized). LT-TCP shows fully utilized bottlenecks and well-managed RED/ECN-controlled queue lengths. Congestion window should be deflated by FEC and packetization overheads to reflect true goodput. However, it does reflect the dramatic reduction in timeouts with LT-TCP over these PER regimes.

3.2 LT-TCP Component Performance

The LT-TCP components are evaluated in the following (cumulative) order:



(a) Congestion Windows (PER = 10%) (b) Congestion Windows (PER = 20%)



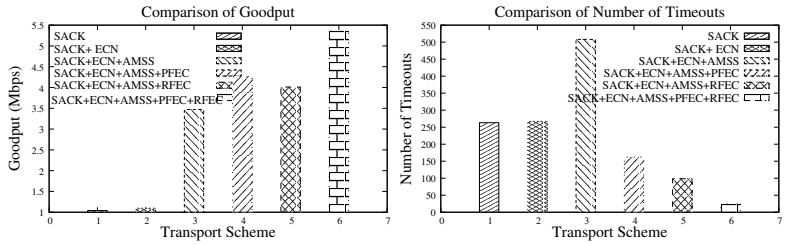
(c) Queuing: SACK (20 %PER) (d) Queuing: LT-TCP (20 % PER)

Fig. 2. Graphs of Congestion Window (in bytes) and Queue Length (in packets) vs Time for TCP-SACK and LT-TCP.

1. TCP-SACK.
2. TCP-SACK with ECN-only (i.e. RED/ECN at bottleneck and congestion response only to ECN marks).
3. TCP-SACK with ECN-only and adaptive MSS.
4. TCP-SACK with ECN-only, adaptive MSS and proactive-FEC (no reactive FEC).
5. TCP-SACK with ECN-only, adaptive MSS and reactive-FEC (no proactive FEC).
6. Full LT-TCP scheme with TCP-SACK, ECN-only, adaptive MSS, proactive and reactive FEC.

The average goodput for the different component bundles is shown in Fig. 3(a). The addition of each component to TCP-SACK consistently improves performance. The final goodput for LT-TCP is over five times the goodput achieved by TCP-SACK. The overhead due to conservative FEC provisioning is reflected in the difference (throughput $= 8.55 Mbps$ vs. goodput $= 5.36 Mbps$). The performance gains of LT-TCP are largely explained through the reduction of timeouts (Fig. 3(b)).

We now examine some of the component-level dynamics. Figure 4 shows the behavior of the adaptive MSS and the resultant effect on congestion window granulation. By varying the MSS with the congestion window, we ensure that a minimum window granulation is maintained, thus increasing the number of



(a) Goodput Comparison (b) Timeout Comparison

Fig. 3. LT-TCP Component Contributions

dupacks and the effectiveness of SACK. MSS also increases when *cwnd* increases, to reduce the packetization overhead.

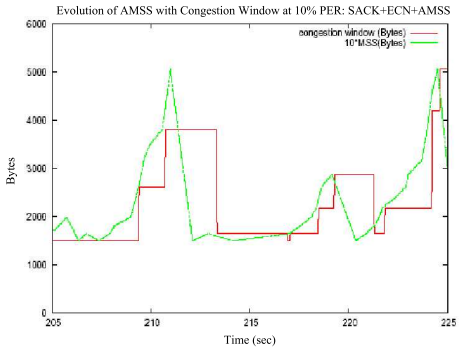
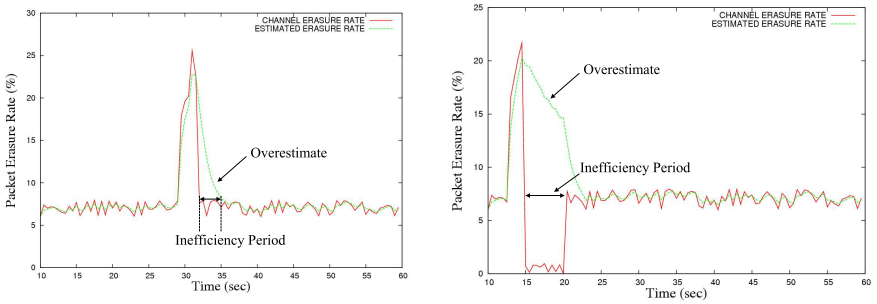


Fig. 4. Effect of Adaptive MSS

Figures 5(a) and (b) illustrates the FEC-estimation and proactive FEC provisioning. We see that estimator (Equation 1) tracks the average erasure rate well, and responds quickly to spikes, but is biased towards over-estimating after the spike vanishes. This overestimate bias will lead to some excess dead-weight FEC, but it has the potential to reduce effects of sudden erasure bursts that can otherwise lead to timeouts.

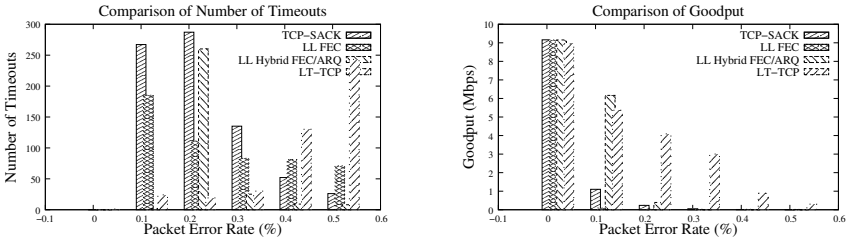
3.3 Comparison with Link-level Schemes

We compare LT-TCP with TCP-SACK and with two other schemes with link layer reliability support: LL-FEC where the link provides FEC to match the average erasure rate on the link, and LL-Hybrid FEC/ARQ where the link provides 10% FEC ($N = 10, K = 9$)and ARQ with a persistence of 3 retries. For LL-FEC, a packet is broken up into N fragments where K units are data units



(a)Transient Response (b) Over-estimation Bias

Fig. 5. Adaptive-Parameter EWMA Erasure Estimator Behavior



(a) Timeout Comparison (b) Goodput Comparison

Fig. 6. Comparing LT-TCP with SACK, LL-FEC and LL Hybrid FEC-ARQ.

that are protected by $R = N - K$ FEC packets. Each fragment is sent independently on the link. For LL Hybrid FEC/ARQ, we added a realistic mix of ARQ persistency (to not impact latency adversely) and FEC protection that does not assume perfect knowledge of the PER on the channel.

Simulation results [3] for number of timeouts and goodput for four schemes is presented in Figure 6. Surprisingly, the LL-FEC scheme does not perform well even at 10% erasure rates and even underperforms SACK (see Table 3 and Figure 6). This is attributable to the deadweight overheads due to short-term mismatches of FEC to erasure rates, even though the long-term average matches the FEC rate.

The LL Hybrid FEC/ARQ with TCP-SACK end-end provides very good performance (Table 4 and Figure 6) when the static FEC protection is matched to the link PER, and backed up by ARQ. However, when the PER exceeds the provisioned FEC value ($> 10\%$), its performance rapidly declines (see Figure 6). Failure to manage timeout risks and limited short-term adaptivity are important reasons for this behavior.

[3] Thanks to Dr. Chadi Barakat, INRIA, whose ns-2 source code and model in [8] we simplified.

	ERROR RATE					
PARAMETER	0 %	10 %	20 %	30 %	40 %	50 %
Goodput(Mbps)	9.15	0.086	0.034	0.02	0.019	0.015
Number of Timeouts	0	185	111	83	81	71
Throughput(Mbps)	9.52	0.1235	0.053	0.03	0.019	0.026

Table 3. SACK + LL-FEC at (0-50%) PER)

	ERROR RATE					
PARAMETER	0 %	10 %	20 %	30 %	40 %	50 %
Goodput(Mbps)	9.15	6.16	0.39	0.002	0.0003	0.0001
Number of Timeouts	0	2	260	25	9	8
Throughput(Mbps)	9.52	6.44	0.49	0.005	0.001	0.0007

Table 4. SACK + LL Hybrid FEC/ARQ at (0-50%) PER

4 Summary and Conclusions

This paper addressed the performance of TCP over networks that include lossy wireless links, where it is well-known that TCP performance suffers substantially when packet erasure rates (PERs) get beyond a small value of about 1% - 5%.

Loss-Tolerant TCP (LT-TCP) contains a complementary set mechanisms for robust TCP performance in ECN-enabled networks, under extreme and highly variable erasure rate conditions (upto 30% PERs). The mechanisms are adaptive and match the amount of error protection and granulation to the conditions of the end-end channel (primarily to reduce retransmissions and avoid timeouts). Thus LT-TCP introduces negligible overhead in the erasure-free case, as one would demand of a transport protocol that needs to operate over wired links that do not have bit errors. LT-TCP does not require any additional router functionality beyond ECN (which has been standardized) As such, it may be easily implemented on an end-to-end basis.

We compared the performance of LT-TCP with TCP-SACK and showed that for the case of 10% packet erasure rate, LT-TCP achieves a factor of 5 better TCP goodput, while introducing about 30% overhead for FEC on the channel at these error rates. We also demonstrate the better performance of LT-TCP compared to link-level FEC and hybrid ARQ/FEC protection. The reasons lie in limited adaptivity at the link layer and inability to avoid TCP timeouts. LT-TCP can complement existing link layer schemes to overcome the residual PER on wireless channels.

We claim that LT-TCP shows consistent and significant *relative* performance improvement for all non-zero erasure rate cases in comparison to the other approaches. However, its *absolute* performance (especially goodput) still suffers for very high erasure rates (40% and higher). Reasons for this clearly lie at the inability to avoid sharply increased timeouts despite the high and adaptive FEC overhead and adaptive granulation policy, and addressing this will be the focus of our immediate future work. In addition, we are examining the following enhancements to the scheme described in this paper.

- Options for better decoupling between the granularity of packets in a window (MSS adjustment) and the amount of proactive FEC added.
- Upon a timeout, ways of dealing with the transmission of the old FEC block of packets at the sender that that are being retransmitted, and decoding of these blocks at the receiver, in relation to the current FEC block of packets.
- At the receiver, when packets are received out-of-order, we deliver only the in-sequence packets to the receiving TCP (unless we know the remaining packets cannot be recovered). This leads to some burstiness in the generation of acks and dupacks. We are exploring ways to mitigate this.
- We are exploring whether or not to generate dupacks upon reception of reactive FEC packets. Further, we need to ensure that reactive FEC packets also honor the TCP window.

We hope to report results of these enhancements and tradeoffs in an upcoming paper.

References

1. D. Aguayo, J. Bicket, S. Biswas, G. Judd and R. Morris, "Link-level Measurements from an 802.11b Mesh Network",*SIGCOMM 2004*, Aug 2004.
2. J. W. Byers, M. Luby, M. Mitzenmacher and A. Rege, "A Digital Fountain Approach to Reliable Distribution of Bulk Data", pp. 56-67, *SIGCOMM 1998*, Aug.-Sep. 1998.
3. Tal Anker, Reuven Cohen and Danny Dolev, "Transport Layer End-to-End Error Correcting", *Leibniz Center, Technical Report*, The School of Computer Science and Engineering , Hebrew University, Jerusalem, Israel, June 2004.
4. J. Nonnenmacher and E. Biersack, "Reliable Multicast: Where to use FEC," *Protocols for High-Speed Networks*, pp. 134-148, 1996.
5. L.Rizzo, "On the feasibility of software FEC", *DEIT Technical Report*, LR-970131, Available at `http://citeseer.ist.psu.edu/rizzo97feasibility.html` .
6. J. Border,M. Kojo, J. Griner, G. Montenegro and Z. Shelby, "Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations",*IETF RFC 3135*, June 2001.
7. C. Barakat and E. Altman, "Bandwidth tradeoff between TCP and link-level FEC", *Computer Networks*, vol. 39, no. 5, pp. 133-150, June 2002.
8. C. Barakat and A. A. Fawal, "Analysis of link-level hybrid FEC/ARQ-SR for wireless links and long-lived TCP traffic", *Performance Evaluation Journal*, vol. 57, no. 4, pp. 423-500, August 2004.
9. C. Casetti, M. Gerla, S. Mascolo, M. Y. Sanadidi, and R. Wang, "TCP Westwood: Bandwidth Estimation for Enhanced Transport over Wireless Links", *Proceedings of ACM Mobicom 2001*, pp 287-297, July 2001.
10. K.K. Ramakrishnan, S. Floyd, and S. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", *IETF RFC 3168*, September 2001.