Disruption-Tolerant Link-level Mechanisms for Extreme Wireless Network Environments

Vijaynarayanan Subramanian,
ECSE Dept, RPI,
subrav@rpi.edu

K.K. Ramakrishnan,
AT&T Labs Research,
kkrama@research.att.com

Shivkumar Kalyanaraman,
ECSE Dept, RPI,
kalyas@rpi.edu

***Abstract*— Wireless links pose significant challenges in terms of achievable goodput and residual loss-rate. Our recent enhancements, called LT-TCP make TCP loss-tolerant in heavy/bursty erasure environments. Link-level protocols mitigate these problems by using a combination of FEC and ARQ but are insufficient when the channel experiences disruptions. When the underlying source of loss is interference (e.g., 802.11 environments), MAC-level mechanisms misinterpret interference as noise leading to poor scheduling (e.g., capture effects) and limit the benefit of transport layer mitigation efforts. We propose enhancements to link-level protocols that enable survival during disruptions. We explore an adaptive link-level strategy to export a small residual loss rate and bounded latency under high loss/ disruption conditions. We evaluate the proposed link-level enhancements, showing that the combination with LT-TCP helps achieve significant end-to-end performance gains. We also demonstrate the trade-off between reduced link layer residual loss (by increasing ARQ persistence) and transport layer timeouts.**

I. Introduction

As the demand for broadband connectivity increases, both cellular and meshed networks will play a role in last-mile wireless distribution networks. Current metro-WiFi deployments (e.g. San Fransisco, Taipei etc) and organic community wireless deployments fit this model. While the performance of TCP/IP has been well studied for one-hop cellular-style low-speed wireless networks, TCP performance over higher-bit-rate, lossy, multi-hop wireless networks is not well understood. It is well-known that wireless links have high, bursty and variable *raw* error rates due to atmospheric conditions, terrestrial obstructions, fast and multi-path fading, active interference and mobility[1]. However, for TCP, what matters is *residual packet erasures* and delay behavior after PHY and LINK layer mitigation has been completed [8]. TCP is exposed to residual error rates which is defined as the error rate subsequent to the link layer's error protection mechanisms.

The rapid deployment of broadband wireless systems such as 802.11 Wireless LANs (WLANs), 802.16 wireless broadband and neighborhood area wireless networks raises expectations of high end-to-end performance. Future cells may be small (around 30 m) in order to deliver high bandwidth to users. This leads to interference among cells in close proximity. Moreover, wireless systems operating in open-spectrum bands will be used in environments characterized by higher levels of interference, capture and disruption phenomena. This is because access points by multiple operators or homes will be set up independently, without global planning or coordination. This phenomena is already apparent in densely populated residential areas, flats and apartments where any client can see a large number of access points, and can interfere with their transmissions during peak usage periods even if the client cannot make an association with the access points (i.e., when access control is enabled). Communication under these demanding conditions will require link and transport layer protocols to handle not only medium access and packet errors, but also handle unexpected periods of capture and disruption due to unavoidable interference. WLAN cells may be vulnerable to *capture effects* that occur due to interference and inability of the MAC layer to fairly distribute transmission opportunities. Disruption periods of several hundred milliseconds are possible under such conditions, especially during peak usage hours. Current link layer protocols are unable to detect and react to such prolonged disruption periods. We demonstrate this issue in the context of 802.11 LANs experiencing co-channel interference.

Currently, TCP-SACK is the most prominent variant of TCP that has been proposed. However, the performance of TCP-SACK is known to fail dramatically beyond an end-end error rate of 5%. Packet losses get mis-interpreted as indications of congestion and the throughput of a connection collapses quickly as the error rate increases. Various mechanisms have been employed at the link-layer to minimize the effect of interference and noise with the aim of reducing the loss rate seen by the transport layer. Automatic Repeat Request (ARQ) and Forward Error Correction (FEC) and combinations of these (Hybrid Schemes) are popular approaches used to reduce the residual error rate on the link.

In this work, we revisit the issue of link-level design with the objective of making it work well under *disruption* channel conditions. In particular, we develop link-layer disruption detection, and respond with a more conservative mode of link-layer hybrid FEC/ARQ scheme (HARQ) and control the *residual error rate* exported to the transport layer. We define *disruption* as conditions under which a packet (and all its fragments when it is fragmented at the link layer) is lost with certainty. We list below the properties that a link-level design should possess. We develop our mechanisms in the context of a generic link layer that suffers from a high packet error rate including disruptions. Our methods can then be customized and integrated within the context of specific MAC layers. Our link-layer objectives include:

Delay Control: Unbounded number of ARQ-style retrans-

This work was funded by AT&T Labs Research, MIT Lincoln Labs/AFOSR Grant Letter No. 14-S-06-0206 and NSF-ITR 0313095.

mission attempts on the link-level is not desirable from an end-end point of view. Link level mechanisms must be able to provide robustness while keeping the link-level latency small. This limits the number of permissible ARQ retransmission attempts.

Small Residual Loss Rate: *Residual loss/error rate*, defined above should be as small as possible so that upper layers such as TCP are exposed to as small an error rate as possible. This must be done however, while keeping the overhead and link latency small.

High Link-Level Goodput: Residual loss rate can be driven to zero, either by trading off latency (more ARQ attempts) or by reducing goodput (by adding FEC indiscriminately). The link HARQ scheme must manage this trade-off because the link-level goodput (discounted by residual loss rate) puts an upper bound on end-to-end goodput.

In-order Delivery: The link-layer should attempt to deliver packets *in-order* since delivery of packets out-of-order to protocols such as TCP will cause unintended side-effects (such as fast recovery).

Limited Impact on Transport Layer: The link level mechanisms should interact minimally with the transport layer to avoid effects such as spurious timeouts, variable end-end RTT and bandwidth. Ideally, the link should provide TCP with the illusion of a *constant bandwidth, zero-loss* pipe. It is challenging to provide this abstraction in multi-hop, lossy and disruption-prone environments.

The contributions of this paper include a) providing insight into the nature of disruption on the link-level and impact on the transport protocol, b) proposing and evaluating a *dynamic* link-level design that can overcome the challenges of operating under such conditions while keeping latency and residual loss rate low and c) investigating the trade-offs between link-level ARQ persistence and transport-level goodput. Moreover, a design goal is to develop a scheme that will work well on all types of wireless links and not just specific systems such as 802.11 links or satellite links.

The rest of the paper is organized as follows. Section II discusses the related work. Section III presents a demonstration of capture effects in 802.11b networks and motivates our work. Section IV provides an overview of the mechanisms proposed. Section V presents the evaluation of the trade-offs between link-level ARQ and transport goodput and the performance of the baseline scheme and proposed enhancements. Section VI summarizes and concludes the paper.

II. Related Work

The menu of error control building blocks is relatively well known: a mix of FEC, ARQ, pipelining, interleaving, packetization, and adaptive modulation/coding (AMC) mechanisms [5], [26]. The open issues for emerging multi-hop and meshed networks operating in extreme environments include: a) how to structure these building blocks into protocols to achieve attractive trade-offs, b) how to divide and balance responsibilities among layers (PHY/LINK/TRANSPORT) to survive a wide range of error characteristics for a broad range of applications, and (c) to understand and minimize the need for cross-layer interactions.

FEC is a popular error mitigation technique for digital voice communications [25], [5]. Bit error correction is usually performed using convolution codes, turbo codes or a mix of coding and modulation [25]. Attractive building blocks for *packet erasure* correction include Reed-Solomon (RS) codes [21], [26], [5] and recently-proposed rateless codes [15]. Unlike FEC, Automatic Repeat reQuest (ARQ) is a closed-loop mechanism that requires error/erasure detection, feedback and retransmission[27], [5]. While FEC suffers from dead-weight overheads in benign conditions, ARQ suffers from delays that worsen in extremely lossy conditions[14]. Hybrids of ARQ and FEC (HARQ) can be created by using FEC in the retransmission phase (e.g. using rate-compatible punctured convolution codes (RCPC), RS-codes [26] or turbo codes[25]), and by jointly decoding over all bits received (e.g. chase combining or incremental redundancy [23], [26], [5]). Modern PHY layers use adaptive modulation in combination with HARQ- or ARQ-based error-control coding [9]. 802.11b MAC offers coarse-grained rate-adaptation in response to changing SINR and packet loss conditions. Regarding packetization, the Radio Link Protocol (RLP) [27], [11] used in cellular systems (IS-95, GSM) fragments packets in smaller units (256-500 bits) to limit the impact of bit errors on packets[27], [19].

Researchers and users constantly push the limits of usage of commodity wireless technology into extreme environments (e.g. an MIT group showed poor link performance (up-to 50% erasure rates) in 802.11b mesh networks [1]). Clearly, the message from prior work is that the potential offered by error control building blocks is yet to be fully realized (especially for operation in extreme environments), and that structuring of such blocks inside protocols and across layers matters.

III. Motivation

In this section, we demonstrate the problem of disruption in IEEE 802.11b systems. Disruption is manifested as *capture effects* in such systems. We show how capture effects are detrimental to transport layer performance and how transport-layer mechanisms assist in mitigating these performance problems.

We consider 802.11b DSSS (2.4- 2.475 GHz using 22 MHz bandwidth) as a baseline wireless system. Since RTS/CTS contention avoidance mechanisms are rarely turned on in practice and even enabling these mechanisms does not guarantee the elimination of interference, we consider a worst-case scenario and assume these mechanisms are turned off. The number of attempts per packet will default to a value of 7 with random exponential back-off used for each retry.

IEEE 802.11b supports four data rates: 1, 2, 5.5, and 11 Mb/s and multi-rate operation to combat slow fading. Every packet, ack or MAC-level ack (MAC-ack) has a preamble of 24 bytes sent at the basic rate 1 Mb/s. The implementation and decision basis to change the rate are usually proprietary though some general heuristics are known [13].

The implicit assumption is that lowering the rate will decrease the probability of packet error. This is true if the causes of packet corruption involve link impairments alone. However, if the cause of packet corruption is *interference*, rate adaptation will not help if the signal strength is high enough. In fact, lowering the rate will expose the packet to higher probability of

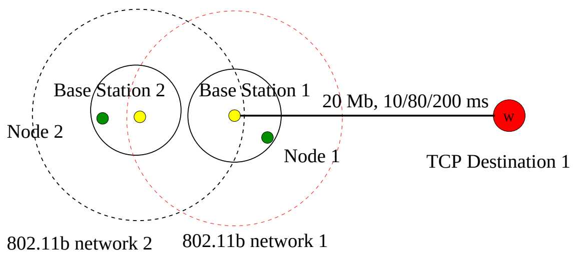


Fig. 1. Simulation Setup for Co-Channel Hidden Node Interference.

error since the packet is "in the air" for a longer time. In other words, rate adaptation is effective in dealing with propagation losses and not with interference losses. In such scenarios, rate adaptation is counter-productive.

A. Co-channel Interference Model: Hidden Nodes in Remote Cells

We assume rate adaptation is turned off and that cells operate at 11 Mb/s. Only the preamble for any MAC transmission is sent at 1 Mb/s. We then examine issues with co-channel interference.

Consider the effect of operating different WiFi cells in close proximity in the same frequency channel. Cells more than one cell-hop away typically reuse the spectrum. As mentioned earlier, due to worst-case design constraints, cells could have radii as low as 30 m. While this design improves SNR when there is no interference, it is detrimental when there is a significant amount of co-channel interference.

The packet corruption due to interference is modeled as follows. While a receiver is receiving a frame, if another transmission occurs in its vicinity and the new transmission's observed signal strength exceeds a threshold at the receiver location, the new transmission corrupts the frame currently being received. It is enough to corrupt a few bits of a packet to render the whole packet useless. However, at high bit rates (11 Mb/s), even 1500 byte packets are short. Further, MAC overheads increase with the number of packets (independent of the size of the packet). Therefore it is better to send larger packets if the bit rate is high (and rate adaptation is turned off).

We assume the transmission range to be 250 m and an interference range to be 500 m. Note that at these ranges, RTS/CTS mechanisms (even if enabled) may not be enough to prevent hidden node interference. The actual patterns of corruption depend upon the relative locations of nodes in cells and patterns of traffic from the interferer and whether the interferer sees reciprocal interference. Moreover, the impact of losing TCP packets vs losing TCP acks is different at the transport layer (acks are cumulative; packets need retransmission).

The simulations were performed using the ns-2 simulator. Six simulations runs were used to obtain each data point of interest. Confidence intervals are shown where applicable.

B. Simulations: Co-Channel Interference (Hidden Node)

We use the scenario shown in Figure 1. There are two cells: Cell 1 and Cell 2, served by base station 1 (BS-1) and base-station 2 (BS-2). Node 2 is *downloading* a file from a server adjacent to base-station 2 (BS-2). This leads to packet transmissions by BS-2 that interfere with BS-1. Assume BS-1 is receiving a large *file upload* from node 1 and relaying it to a remote server (which could be 5ms, 40 ms or 100 ms away). Therefore, BS-1's receptions suffer from corruption due to interference. Since BS-1's transmission of TCP acks or MAC acks are short, and it only interferes with BS-2's reception of short TCP acks or short MAC acks (which can be recovered with MAC-level ARQ), there is little effect on the download performance seen by node 2. Further, since node 2 sees a short RTT, it ramps up its window faster and essentially "captures" the channel for a period of 250 ms.

Node-1's upload session is effectively shut out for 250 ms every 2 seconds. During this period, each packet at node 1's queue is given to the MAC layer which attempts back-off and retransmission 7 times (roughly 60 ms per packet) before dropping the packet. The TCP layer will see a pattern of no residual loss during periods of no-interference and a huge burst loss during the capture period. In addition, a queue builds up at node 1's IP layer since the MAC layer takes longer to transmit each packet during capture. We therefore recommend careful buffer size settings and conservative RED thresholds to absorb this sudden burstiness and accommodate a larger window to tolerate capture. We will see that LT-TCP's adaptive MSS method will granulate the window to reduce the likelihood that an entire window is lost during capture and that reactive recovery mechanisms work.

Our first set of results (Table I) compares TCP-SACK and LT-TCP performance when there is no interference (i.e., Cell 2 is quiet). We vary the RTT to be 10ms, 80ms and 200 ms. These numbers are representative of modes in observed RTT distributions reported by CAIDA's Skitter measurement project [10]. The short RTT (10 ms) represents intra-metro or intra-regional RTT (e.g., within the Bay area); medium RTTs (80 ms) represents US east-west coast RTTs; and 200ms (and higher) RTTs are observed in transcontinental links (between US, Europe or Asia). The reason we examine multiple RTTs is because even though the WiFi link itself is a LAN link, the end-to-end RTT matters for TCP-SACK when there is even a small residual erasure rate.

As expected, the goodputs seen by TCP-SACK and LT-TCP are comparable (4.4-4.6 Mb/s) and are close to the maximum possible on 802.11b links with no rate adaptation, and MAC-acks sent at 11 Mb/s regardless of RTT.

In the second set of results (Table II), we use ARQ = 7 (i.e., six retransmissions at the MAC layer at 11 Mb/s) with 250 ms interference/capture every 2 seconds. Due to exponential back-off, these six retransmissions take upto 60-75ms before a packet is dropped during the capture phase. TCP-SACK goodput improves for both the LAN (10 ms RTT) and USA continental WAN (80 ms RTT) case, although it still collapses for longer RTTs due to high sensitivity to residual error rates. LT-TCP's performance is competitive with TCP-SACK for LANs, and is clearly superior for longer RTTs. This set of results suggests that link level ARQ is not a panacea even with LAN links because the end-to-end RTT still matters. Moreover, such high degrees of ARQ persistence are not possible for longer delay links such as satellite links, which supports the case for end-to-end mechanisms like LT-TCP.

Thus, we observe that PHY and MAC layer mechanisms have adaptation techniques designed primarily to handle channel im-

PARAMETER	LT-TCP			TCP-SACK		
RTT	10ms	80ms	200ms	10ms	80ms	200ms
Goodput(Mb/s)	**4.43**	**4.40**	**4.39**	**4.64**	**4.63**	**4.52**
95% CI for Good-put	[4.36,4.49]	[4.34,4.46]	[4.34,4.43]	[4.61,4.62]	[4.63,4.65]	[4.45,4.61]
Number of Timeouts	0	0	0	0	0	0
MAC Throughput(Mb/s)	5.70	5.68	5.64	5.89	5.88	5.72

TABLE I

No interference: LT-TCP and TCP-SACK performance without interference under various conditions of end-end delay.

PARAMETER	LT-TCP			TCP-SACK		
RTT	10ms	80ms	200ms	10ms	80ms	200ms
Goodput(Mb/s)	**3.72**	**3.76**	**2.54**	**4.08**	**3.07**	**0.37**
95% CI for Good-put	[3.70,3.74]	[3.69,3.83]	[2.43,2.64]	[4.07,4.09]	[2.98,3.15]	[0.3,0.44]
Average Number of Timeouts	0	0	0	0	0	25.8
MAC Throughput(Mb/s)	5.24	5.26	3.56	5.44	4.00	0.62

TABLE II

ARQ = 7, 250 ms / 2 s interference :LT-TCP and TCP-SACK performance with interference of 0.25 seconds out of 2 seconds under conditions of varying end-end delay.

pairments (e.g., rate adaptation, low rate preamble, low-rate control packets like MAC-acks) and export a relatively "clean" virtual link to higher layers. However, these PHY-level adaptive/modulation coding (AMC) or rate adaptation techniques tend to not be appropriate when the primary source of corruption is interference. Such techniques confuse interference as noise (somewhat akin to transport layer mechanisms confusing packet erasure as congestion). Aggressive PHY rate-adaptation response in such situations is counter-productive because the packets are "on-the-air" longer resulting in exacerbating the interference problem. Moreover, it also eliminates possibilities of mitigation at higher layers (link or transport). Based on these observations, we examine possible link-layer mechanisms to overcome such disruptions. We broaden our study to examine wireless channel disruptions in general, going beyond the issue of co-channel interference.

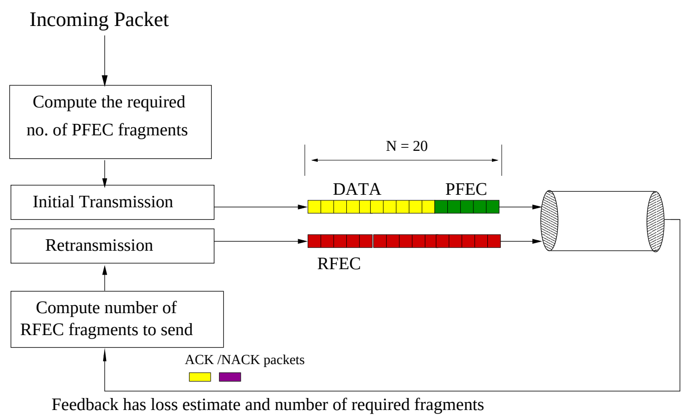


Fig. 2. Protocol operation over a lossy channel: The initial data+PFEC transmission leads to feedback. If insufficient number of units reached the receiver, RFEC units are sent in subsequent attempts to recover the packet.

IV. Link-level Protocol

In this section, we present the basic link-level protocol and discuss the building blocks used to construct it.

A. *Basic Hybrid FEC/ARQ Link-level Protocol*

We now look at the link-level mechanisms that comprise our hybrid FEC/ARQ scheme. As seen in Figure 2, each set of K data units (fragments at link-layer) is protected with the addition of $N - K$ proactive FEC (PFEC) units to create a block of size N. The amount of proactive protection that is provided is a function of the current estimate of the loss rate. If less than K units arrive uncorrupted at the receiver, N reactive (RFEC) FEC units are sent to make up for the missing units in the retransmission phase(s). Due to the sequence-agnostic property of FEC, the receiver needs to receive *any* K units (of data or PFEC or RFEC) to reconstruct the original K data units. We compute FEC using a method similar to that used in CD-ROMs, i.e., shortened Reed-Solomon (R-S) codes which incur storage. Alternatively, one could avoid storage and use the Fountain codes or Raptor codes [7], [24] to compute erasure correction units on demand. The link layer protocol relies on the building blocks of loss estimation, adaptive fragmentation, FEC provisioned as proactive and reactive FEC. The sender also uses information sent in the feedback (through ACK/NACK packets). These building blocks are described below.

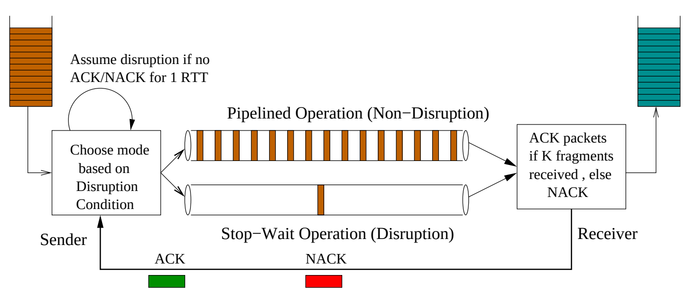


Fig. 3. Overview of the link level mechanisms: Disruption is detected by the sender and packets are sent in *stop-wait mode*. Upon exiting disruption mode, packets are sent again in *pipelined* mode.

Link Layer Loss Estimation: We use per-frame loss rate sampling, i.e., the sample is the ratio of number of units lost divided by the total number of units sent. The sample includes

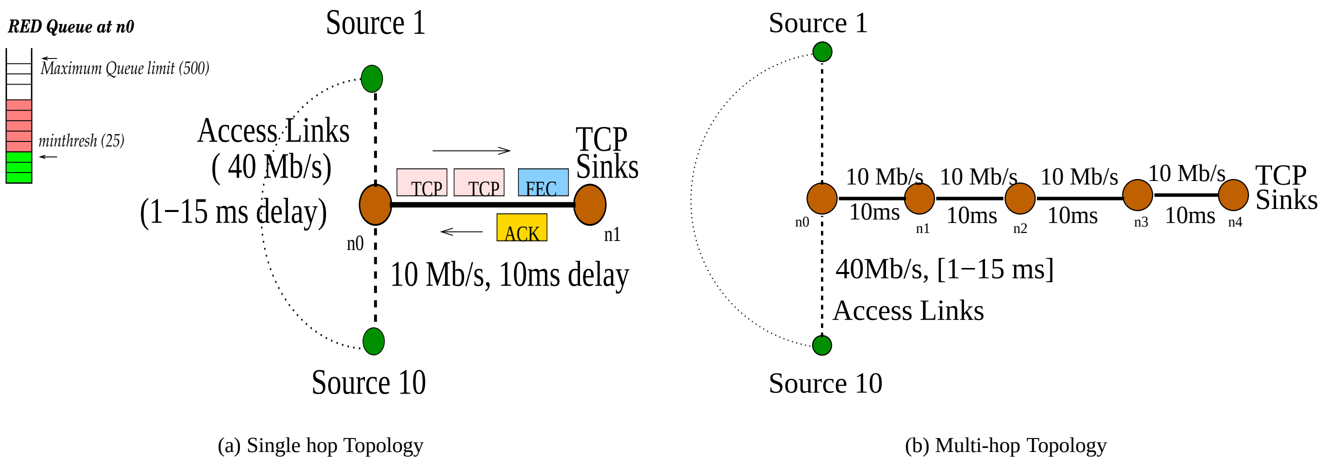


Fig. 4. Test Configurations: 1-Lossy Link Case and Multi-Hop Path Case. Each link is affected by the disruption error process as described.

both data and FEC packets with the estimate being obtained separately for the Proactive FEC and Reactive FEC phases. An Exponential Weighted Moving Average (EWMA) (with $\alpha = 0.005$) is used to estimate the loss rate from these samples.

Link Layer Packetization: A single link-layer frame is chosen to be the FEC block. The frame (data + PFEC units) is fragmented into units ($N = 20$) which are subject to potential erasure. The size of each unit is finalized once the amount of proactive FEC (PFEC) units per-frame is determined (see below). With $N = 20$ units/frame, this policy limits PFEC to be provisioned in multiples of 5%.

Link Layer PFEC: The number of PFEC units sent along with the data units in the original transmission is computed based on the current estimate of the loss rate. We set PFEC protection to one standard deviation more than the expected loss, thereby providing some protection against underlying variance in the loss process. This is even more important under conditions of disruption.

Link Layer RFEC: To achieve the goal of limited ARQ persistence, high goodput and low residual loss, we use an aggressive RFEC strategy where the number of RFEC units sent in the retransmission phase(s) is set to $N = 20$.

Link Layer Feedback Design: Feedback at the link layer encodes the particular lost frame unrecoverable with just PFEC Since units are assumed to arrive in-order in the best case scenario, out-of-order detection of units belonging to the next frame trigger the feedback. Frames are only delivered error free and in-order at the link-receiver.

B. Disruption-Mitigation Enhancements

We now look at the specific enhancements to the basic protocol described above that are designed to mitigate disruptions. The key mechanisms are:

Modal Operation: The sender can operate both in *stop-wait* mode (where each packet has to be acked/nacked before the next packet can be sent) and in *pipelined* mode (where a window of packets can be sent to fill the pipe). Operating in *pipelined* mode increases achievable throughput/ goodput but can erase several packets if an disruption event occurs.

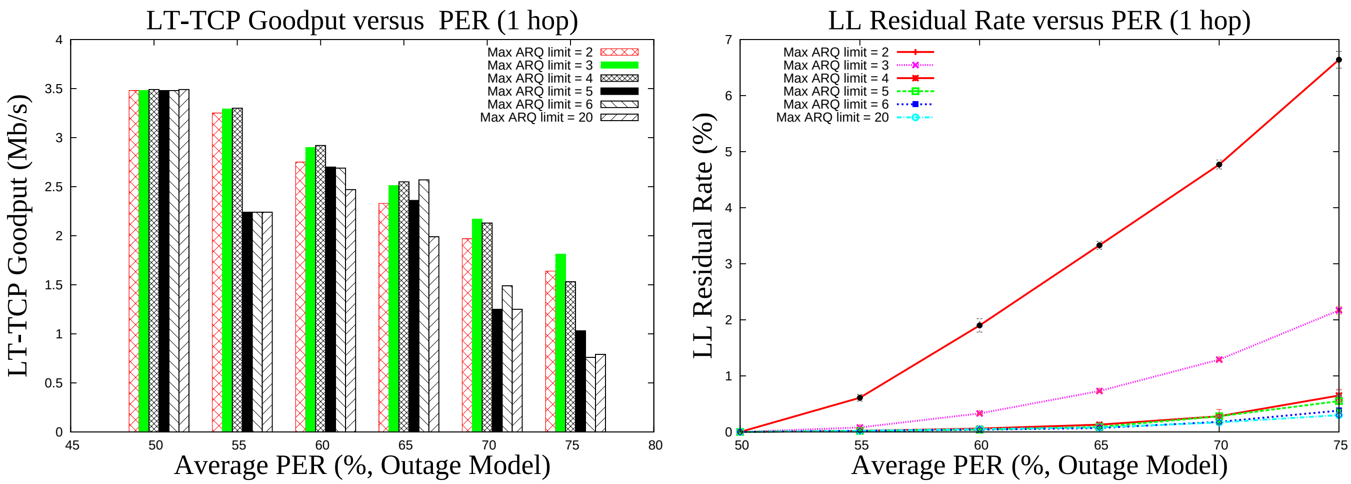
Disruption Detection: The receiver is capable of detecting disruption based on a simple heuristic. In our scheme, we assume that if all the fragments of x conservative packets are corrupted, the channel is in disruption. We set $x = 1$ to detect disruption aggressively.

Retransmission Strategy: A packet can be sent at most twice in *pipelined* mode. Transmission attempts made in *stop-wait* mode are not counted as true transmissions. However, to keep the link-delay bounded the ARQ persistence must be small. The choice of the optimal ARQ persistence is dependent on the length of disruption periods. On the other hand, to overcome disruption periods on the order of around 100 ms, ARQ persistence cannot be too small. To meet these goals, we choose a reasonable value for the limit on the ARQ persistence of 3, irrespective of the mode of transmission. Thus, a packet is discarded after 3 retransmission attempts beyond the original transmission. For the disruption time-scales examined here, this choice is borne out in the performance evaluation in Section V-B.

Fig 3 illustrates the operation of the disruption detection and response mechanisms and shows the dual mode of operation. The sender chooses the mode of operation based on its detection of disruption. Feedback from the receiver indicates whether the packet was received correctly. If additional units are needed to decode a packet correctly, the feedback indicates this. The figure also shows how the mode of operation is chosen.

V. Performance Results

In this section, we look at the performance of a baseline scheme and study the trade-offs between ARQ persistence and its impact on TCP-level performance. We then present the performance of the link-level disruption mitigation enhancements in conjunction with TCP-SACK and LT-TCP. We study the impact both on single hop and on 4-hop topologies. We use a single-bottleneck test case (see Fig. 4: 10 Mb/s bottleneck, 10 ms one-way delay, 10 TCP flows) under the disruption error model described below. Hosts are ECN-enabled, implementing RED/ECN and the $minthresh$ and $maxthresh$ values are shown in the figure. The simulations were run for 200 seconds, and results are averaged over 10 randomized runs. Confidence intervals are shown as applicable. We first present the loss / error model used in the performance evaluation.



(a) (b)

Fig. 6. LT-TCP: Single hop, 10 flows , Transport goodput obtained with different ARQ persistence and the corresponding link-layer residual loss rates.

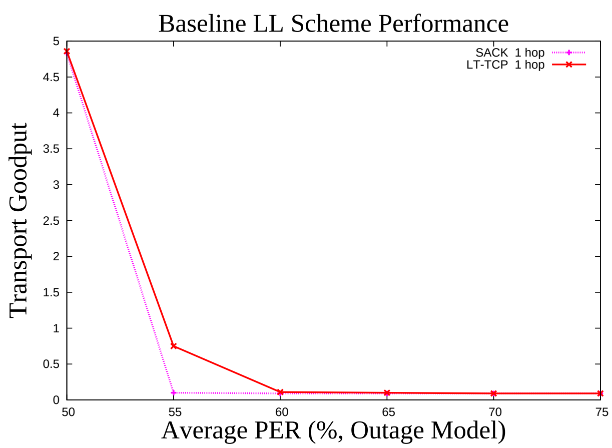


Fig. 5. Transport-layer goodput obtained with TCP-SACK and LT-TCP in conjunction with the baseline link-layer scheme. Without disruption mode enhancements and FEC support, performance drops dramatically even with high ARQ persistence.

Disruption Error Process: We consider a two-state disruption loss/ error process with ON and OFF periods where the error rate is applied at the granularity of a packet. The packet error rate (PER) in the ON state is 100% i.e. any packet in this state is erased. In the OFF state, the PER varies from 0-50%. The average PER therefore varies from 50% to 75%. In our simulations, the bursty loss model has ON/OFF periods with a mean of 100 ms, randomized over a small range (9-11ms). To avoid the effect of transients caused by losing the entire initial window of TCP, the initial 15 seconds of the simulation are assumed to be error-free.

A. Baseline Scheme

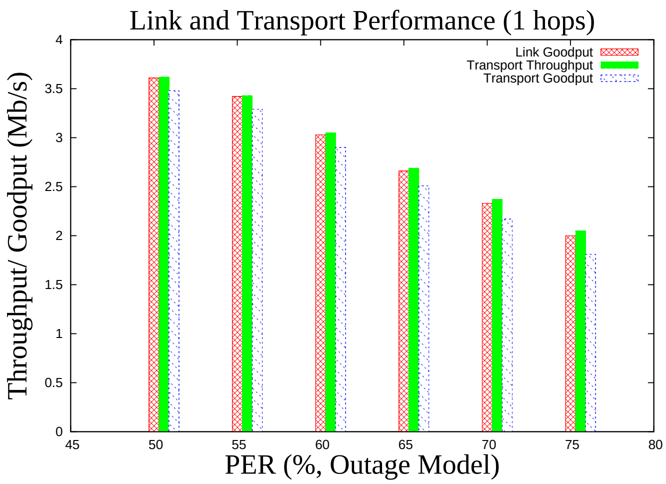
We first consider a baseline link-layer scheme where the link-layer scheme operates only in *pipelined* mode (i.e., no disruption detection and consequent mode-switching), with a high ARQ limit of 7 (such as is typical in 802.11b networks) and has no FEC support at the link layer. Under these conditions, we look at the performance of TCP-SACK and LT-TCP. The transport-layer goodput obtained is as shown in Fig 5. As we expect, the transport layer goodput drops very rapidly as the link error rates go up, and even the improvements with LT-TCP are unable to recover from the severity of the effect of disruptions.

B. Trade-off between Link ARQ Persistence and End-End Performance

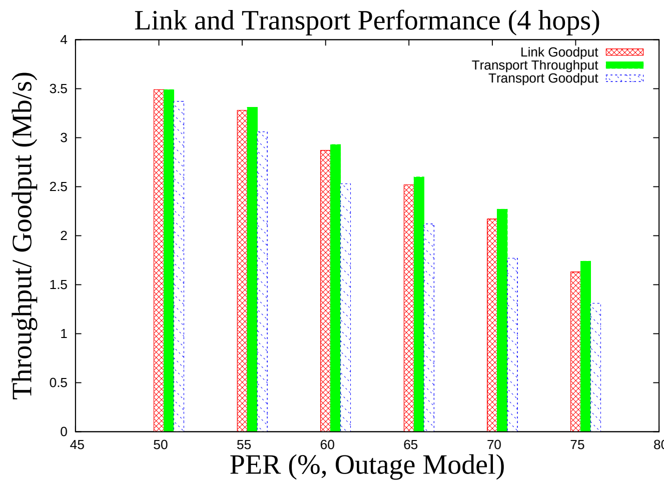
We now examine the impact of increasing the ARQ persistence on the link residual loss rate and the end-end TCP goodput with LT-TCP. We consider the environment of a single lossy wireless hop with 10 end-to-end transport flows. The link suffers from a *disruption* loss process as described earlier for the underlying packet erasure rate, varying from 0 to 50%.

The trade-off is that an increased link-layer ARQ persistence has the potential to reduce the link residual packet erasure rate, at the expense of additional delay. Our model for ARQ retransmissions is somewhat optimistic in that we do not incorporate a backoff for the timer between successive retransmission attempts. The additional delay due to ARQ retransmits can potentially cause the end-to-end TCP layer to timeout, thus causing a degradation in the overall goodput.

We consider maximum ARQ limits of 2,3,4,5,6 and 20. Figure 6(b) shows the residual link PER as the underlying PER varies. We observe that there is a significant benefit from increasing the ARQ limit from 2 through 4. However, increasing the limit further to 5 and beyond shows that we have reached a point of diminishing returns. Figure 6(a) shows that at higher PERs, the LT-TCP goodput first increases as the ARQ limit increases, but then starts to reduce. The initial increase in the goodput for ARQ limit going from 2 to 3 (and in some cases 4) is because of the reduction in the residual link PER (thus providing a better link to the transport). However, when the ARQ limit goes to 4 and above, the LT-TCP goodput starts dropping rapidly. We find that the number of timeouts experienced by the TCP connections (in some cases, these are in fact coincident among the multiple flows, thus idling the link) increases as the ARQ limit goes up to 4 and above. Thus, the reduction in the residual link PER comes at the price of higher delay and resulting timeouts at the TCP layer. We expect this to be



(a) 1 hop Scenario



(b) 4 hop Scenario

Fig. 7. LT-TCP Performance: Link and Transport throughput and goodput with disruption mitigation in the Link Layer: single-hop and 4-hop scenarios.

exacerbated if we were to accurately model the backoff of successive retransmission intervals, as is true with MAC protocols such as 802.11b. We see that to be able to operate under high underlying PER on the link, having an ARQ limit of 3 strikes a balance between reducing the residual link PER and maintaining high TCP (LT-TCP) goodput without experiencing the penalty of frequent transport timeouts. Successive timeouts experienced at the TCP layer have an even more serious effect due to the potential for significant idling of the link due to the TCP retransmit timer also backing off substantially. In the remaining simulations, we assume a ARQ limit of 3. Moreover, this reasonable limit on ARQ persistence keeps the link-level delay low and bounded which translates to lower end-end delay.

C. Single and Multi-hop Scenarios

We now look at the performance of TCP-SACK and LT-TCP operating in conjunction with the link-level enhancements over 1-hop and 4-hop scenarios.

Figures 7(a) and 7(b) show the link-level goodput and transport-level throughput and goodput for the 2 scenarios. It can be seen that in the worst case where we have 4 hops with an average error rate of 75%, the transport-level goodput is still above 1 Mb/s. In the single hop case, the link-goodput is close to the transport-throughput as the residual loss rate on the link is low. Over 4 hops, residual loss rates increase (especially at higher average PER) and the difference between the transport layer throughput and link-layer goodput increases.

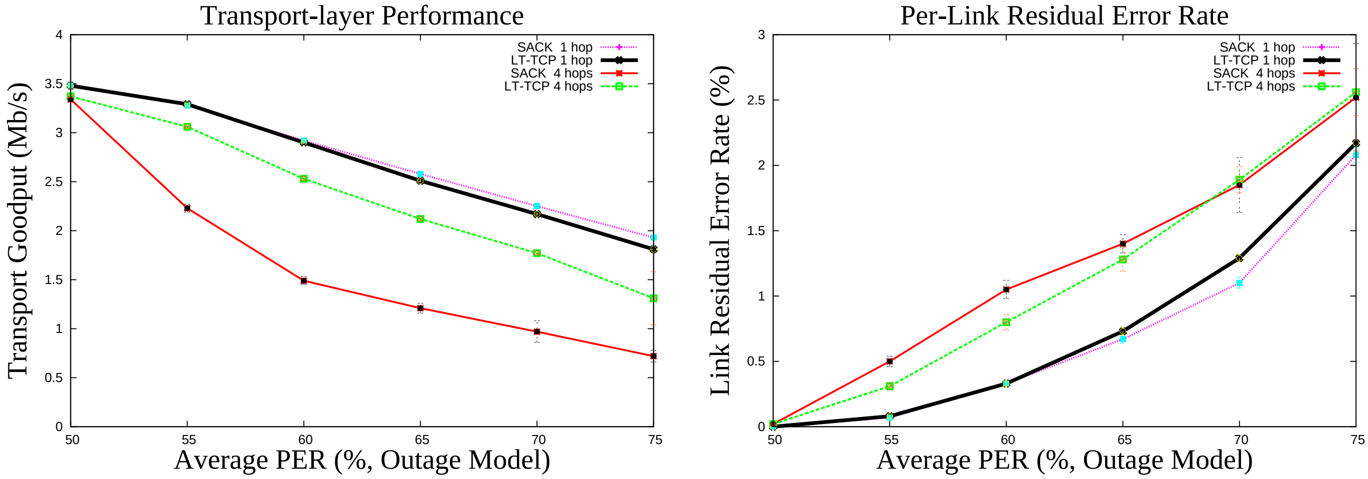
Figure 8(a) shows the transport-level goodput (for TCP-SACK and LT-TCP). With a single hop scenario, the residual error rate is low ($< 5\%$). As noted earlier, TCP-SACK performance degrades only beyond an error rate of 5%. Since, LT-TCP incurs higher packetization overhead due to its adaptive segmentation strategy (on small per-flow bandwidth paths), its performance is slightly lower than that of TCP-SACK. However, as the number of hops increases to 4, the end-end loss rate is significant enough to cause a breakdown in TCP-SACK's performance. The combination of LT-TCP and link layer enhancements however is able to perform well. The improved performance is partly explained by the reduction in the number of timeouts with LT-TCP (5 compared to 25 for TCP-SACK). This also indicates that even over 4 hops, the link-level enhancements interfere minimally (and complement) with the transport mechanisms utilized in LT-TCP.

Figure 8(b) shows the per-hop residual loss rate for the same scenarios. For the 4-hop scenarios, the residual loss rates accumulate, leading to significant end-end loss rate. TCP-SACK is unable to perform well at these high end-end loss rates whereas the combination of the link-level enhancements and LT-TCP is able to deliver higher transport-layer goodput.

VI. Summary and Conclusions

This paper addressed the performance problems experienced by transport protocols over network paths that include lossy links. It is well known that TCP performance degrades sharply beyond an end-end loss rate of 5%. We have proposed a highly loss-tolerant TCP protocol (LT-TCP) using an adaptive, end-to-end hybrid ARQ/FEC reliability strategy exploiting ECN for congestion detection. To complement this transport protocol, we proposed a set of link-level enhancements that meets the objectives of exporting a low residual loss rate, maintains low latency and high link goodput even under scenarios when the link experiences disruptions.

We first looked at an example of disruption in 802.11 networks and saw how capture effects can be detrimental to performance. In this situation, our proposed transport-layer enhancements (LT-TCP) helps improve performance, especially over longer RTTs. We then considered an abstract, lossy link and developed a link-level mechanisms designed to meet the above goals. In particular, we considered the problem of disruptions and proposed the techniques of *disruption-detection* and *mode-switching* to combat it. We demonstrated the trade-off between higher persistence at the link-level and reduced goodput at the transport layer (as a consequence of timeouts) and argued for an ARQ limit that provides a good balance between the two. Over



(a)

(b)

Fig. 8. TCP-SACK and LT-TCP goodput over 1 and 4 hops and corresponding per-hop residual loss rate with link-layer disruption mitigation enhancements.

multi-hop paths, we showed how the residual loss rates can accumulate. This causes TCP-SACK, even with our link-layer enhancements, to perform poorly. The support provided by the LT-TCP enhancements to make transport layer robust to losses overcomes this performance penalty. Our results showed the superior performance of the combination of LT-TCP and our proposed link layer enhancements. We showed that it is possible to provide a link with significantly reduced delay (under loss and disruptions), high goodput and low residual loss rate and in-order delivery. Such a link introduces minimal interference with TCP mechanisms even under extreme conditions such as disruptions.

In the future, we plan to explore the issue of disruptions when the period of disruption is comparable or smaller than the link-level RTT. These conditions are more challenging since disruption-detection cannot be easily performed within the short timescales.

References

[1] D. Aguayo, J. Bicket, S. Biswas, G. Judd, and R. Morris. Link-level Measurements From an 802.11b Mesh Network. *SIGCOMM Computer. Communications. Review.*, 34(4):121–132, 2004.

[2] T. Anker, R. Cohen, and D. Dolev. Transport Layer End-to-End Error Correcting. Technical report, The School of Computer Science and Engineering , Hebrew University, 2004.

[3] C. Barakat and E. Altman. Bandwidth Tradeoff Between TCP and Link-level FEC. *Computer Networks*, 39(2):133–150, June 2002.

[4] C. Barakat and A. A. Fawal. Analysis of Link-level Hybrid FEC/ARQ-SR for Wireless Links and Long-lived TCP Traffi c. *Perform. Eval.*, 57(4):453–476, 2004.

[5] E. Berlekamp, R. Peile, and S. Pope. The Application of Error Control to Communications. *Communications Magazine, IEEE*, 25(4), April 1987.

[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] J. W. Byers, M. Luby, M. Mitzenmacher, and A. Rege. A Digital Fountain Approach to Reliable Distribution of Bulk Data. In *SIGCOMM*, pages 56–67, Aug-Sep 1998.

[8] S. Dawkins, G. Montenegro, M. Kojo, and V. Magret. End-to-end Performance Implications of Slow Links. IETF RFC 3150, July 2001.

[9] A. Goldsmith. *Wireless Communications*. Cambridge University Press, 2005.

[10] B. Huffake, M. Fomenkov, D. Moore, and K. Claffy. Macroscopic Analyses of the Infrastructure: Measurement and Visualization of Internet Connectivity and Performance. URL:http://www.caida.org/.

[11] H. Inamura, R. Ludwig, and F. Khafi zov. TCP Over Second (2.5G) and Third (3G) Generation Wireless Networks. IETF RFC 3481, February 2003.

[12] L. Baldantoni and H. Lundqvist and G. Karlsson. Adaptive End-to-End FEC for Improving TCP Performance over Wireless Links. In *ICC*, June 2004.

[13] M. Lacage, M.H. Manshaei, and T. Turletti. IEEE 802.11 Rate Adaptation: A Practical Approach. In *Proceedings of the 7th ACM International Workshop on Modeling, Analysis and Simulation of Wireless Systems*, pages 126–134, Venice, Italy, 2004. ACM Press.

[14] P. Lettieri, C. Fragouli, and M. Srivastava. Low Power Error Control for Wireless Links. In *MobiCom '97: Proceedings of the 3rd annual ACM/IEEE international conference on Mobile computing and networking*, pages 139–150, New York, NY, USA, 1997. ACM Press.

[15] M. Luby. LT-Codes. In *Proceedings of the 43rd Annual IEEE Symposium on the Foundations of Computer Science (STOC)*, pages 271–280, November 2002.

[16] S. Mascolo, C. Casetti, M. Gerla, M. Y. Sanadidi, and R. Wang. TCP Westwood: Bandwidth Estimation for Enhanced Transport over Wireless Links. In *Mobile Computing and Networking*, pages 287–297, 2001.

[17] J. Nonnenmacher and E. Biersack. Reliable Multicast: Where to Use FEC. In *Protocols for High-Speed Networks*, pages 134–148, 1996.

[18] O. Tickoo and V. Subramanian and S. Kalyanaraman and K.K.Ramakrishnan. LT-TCP: End-to-End Framework to Improve TCP Performance over Networks with Lossy Channels. In *Thirteenth International Workshop on Quality of Service (IWQoS 2005)*, Passau, Germany, June 2005.

[19] A. Konrad R. Ludwig and A. D. Joseph. Optimizing the End-to-End Performance of Reliable Flows over Wireless Links. In *MOBICOM*, August 1999.

[20] K.K. Ramakrishnan, S. Floyd, and D. Black. RFC 3168 - The Addition of Explicit Congestion Notifi cation (ECN) to IP, Sep 2001.

[21] I. Reed and G. Solomon. Polynomial Codes Over Certain Finite Fields. *Journal of the Society for Industrial and Applied Mathematics*, 8:300–304, June 1960.

[22] L. Rizzo. On the Feasibility of Software FEC. DEIT Technical Report LR-970131. Available as http://www.iet.unipi.it/ luigi/softfec.ps.

[23] D. N. Rowitch and L. B. Milstein. On the Performance of Hybrid FEC/ARQ Systems Using Rate Compatible Punctured Turbo (RCPT) Codes. *IEEE Transactions on Communications*, 48(6):948–959, June 2000.

[24] A. Shokrollahi. Raptor Codes. Technical report, Digital Fountain, Inc, June 2003. Tech. Rep. DF2003-06-001,.

[25] B. Sklar. *Digital Communications, Fundamentals and Applications*. Prentice Hall, 2 edition, 2001.

[26] S. B. Wicker. *Error Control Systems for Digital Communication and Storage*. Prentice Hall, 1 edition, 1995.

[27] G. Xylomenos and G.C Polyzos. Multi-service Link Layer Enhancements for the Wireless Internet. In *Computers and Communication, 2003. (ISCC 2003). Proceedings. Eighth IEEE International Symposium on*, volume 2, pages 1147 – 1152, 2003.