Experimental Study of Link and Transport Protocols in Interference-Prone Wireless LAN Environments

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

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

Shivkumar Kalyanaraman,
IBM India Research Labs
shivkumar-k@in.ibm.com

***Abstract*—As wireless networks are deployed widely and user expectations grow, it is important to study the trade-offs among goodput, latency and loss rates that exist in link and transport protocols. Under low residual loss rates, retransmissions can be effective in bringing down the loss rate. However, at high loss rates, current links trade-off goodput and low latency for a small loss rate which diminishes overall performance. We look at the performance at the link and transport layers in the presence of raw loss and see how current protocols make an unattractive trade-off at the link layer. We also show high ARQ on the link can cause high per-packet latency on the link which can lead to interactions with TCP mechanisms leading to spurious retransmissions and timeouts and reduced TCP-layer goodput. Our measurements results, based on experiments conducted on the ORBIT testbed, show the need for modifications to the link layer to obtain a favorable three-way trade-off. In prior work, we have developed link and transport protocols that are designed to work under high loss rates. Our link design LL-HARQ (hybrid FEC/ARQ) is designed to minimize the link delay while exporting a very small residual loss rate and high goodput while our transport protocol Loss-Tolerant TCP is designed to operate over a wide range of residual loss rates. We provide insights about the balance between error-protection functions at the two layers and examine the case for cross-layer co-operation. The measurement traces were used as inputs to our simulations on the ns-2 simulator. Our results show the favorable trade-offs among goodput/latency/loss rate obtained with our approach over traditional approaches.**

I. Introduction

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. 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 planned deployments (e.g. San Fransisco, Google Wifi, AT&T Metro Wifi etc.) and organic community wireless deployments fit this model. 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[2]. However, for TCP, what matters is *residual packet erasures* and delay behavior after PHY and LINK layer mitigation has been completed. TCP is exposed to residual error rates which is defined as the error rate *subsequent* to the link layer's error protection mechanisms.

Our goal is to evaluate the three-way trade-off between link latency, residual link loss rate and link goodput in the presence of packet errors. Under conditions of low loss, packets on the link layer would incur limited latency impact since reliability techniques such as stop-and-wait will be sufficient to recover from link losses. This is especially true for smaller delay wireless LANs where the delays will be of the order of a few milliseconds. However, as we move from small delay last-hop links to wireless backhaul links and links that are part of a multi-hop path such as a wireless mesh, we find that the trade-offs made by current link protocols relying on ARQ persistence are sub-optimal. As the packet error rate increases, the number of ARQ retries needed to export a small (ideally zero) loss rate to the higher layer increases. While ARQ with higher and higher degrees of persistence do indeed lead to lowered residual loss rates, they present poor trade-offs on the other dimensions: goodput and (to a lesser/indirect extent) latency. This poor trade-off may matter less in single-hop wireless links but the latency effects are exacerbated in metropolitan and wide-area networks where the links are required to provide a high data rate.

We investigate the nature of real wireless links by conducting experiments on Open-Access Research Testbed for Next-Generation Wireless Networks (ORBIT)[1]. We use the ORBIT experimental traces to demonstrate that a well-designed combination of link and transport layer HARQ can harness a significant portion of theoretical capacity (and trade-off some for latency benefits). In prior work[12], we have developed enhancements to the standard protocols at the link and transport layers. These protocols have the same guiding design principles and are designed to operate independently. The transport layer called Loss-Tolerant TCP (LT-TCP) is robust at high loss rates (up to average error rate of 50%). At the link-layer, we have developed a protocol called LL-HARQ which exports a favorable trade-off between goodput, latency and residual loss rate to the upper layers. We have shown in earlier work that LL-HARQ can obtain a very good three-way trade-off with just one retransmission attempt. One of the motivations is the realization that there is high theoretical fraction of raw capacity available on a link to be harnessed despite high raw loss rates which is not realized by standard protocols.

High ARQ persistence at the link-layer leads to delay spikes and variable round-trip times which can cause negative interactions with TCP. While link-level support with low ARQ (such as in LL-HARQ) can decrease the link latency, the small residual loss rate exported can aggregate over multiple hops to present TCP with a significant end-to-end loss rate. We provide insights into the structuring of the building blocks and balance between error-protection functions at the two layers and examine the case for cross-layer co-operation. We demonstrate that the combination achieves improved performance (delay, loss and goodput) over traditional approaches.

Our objectives in this paper are as follows:

1) To study the nature of wireless losses in the presence of varying levels of interference (modeled by injecting noise at the receiver). We wish to see the impact of noise on raw link loss rates, residual loss rate experienced by the transport layer and the throughput/goodput obtained at each layer and the overheads and penalties incurred.
2) To study the impact of high ARQ (retransmissions) at the link layer on the raw and residual loss rates. We can then see the impact of the loss rates on the link and transport goodput.
3) To study the impact of high latency/delay spikes and the interactions between the link and transport layers. While high ARQ reduces the residual loss rates, the link latency increases. The latencies experienced on the link can not only be high but also variable. This can interact negatively with transport layer mechanisms. Delay and latency spikes can lead to spurious timeouts at TCP [10] which can have the effect of keeping the link idle leading to lower than expected throughput/goodput.
4) Finally, we use the data traces gathered to come up with realistic link error models. These models are used as input to the ns-2 simulator wherein we test our proposed and the standard protocols. Our data traces and error models provide a level of realism in addition to the synthetic error models we use to stress-test our simulations.

Apart from the measurements, our research contributions include demonstrating and verifying the negative impact of high link ARQ on TCP performance (in terms of goodput) and interactions with transport retransmissions leading to spurious timeouts. Our link measurements on ORBIT yield traces and loss models that we use to test our proposed protocols (LT-TCP at the transport layer and LL-HARQ at the link layer) against standard protocols. We demonstrate the efficacy of our proposed solutions using these realistic link models. We then extend these 1-hop link traces to test our protocols under more stressful multihop scenarios.

The rest of the paper is organized as follows. Section II discusses some of the related work in the area of wireless measurements and protocol development. Section III details the experimental setup on the ORBIT testbed and our results and insights from the measurements. Section V presents an overview of the link and transport protocols that were developed in prior work. Section VI tests the proposed protocols using the ns-2 simulator both using trace-driven simulations as well as with more stressful synthetic error models. Section VII concludes the paper.

II. Related Work

Network designers have known that what matters to end-to-end protocols is the residual packet erasure characteristics of wireless links after any link-layer error mitigation is completed [6]. However, the situation with current standard link-level mechanisms is not encouraging. A performance study on long-range 802.11 wireless links [5] showed that the link packet error rate can vary rapidly between 0% and 100% when the received signal to-noise ratio (SNR) changes by as little as 4-6 dB. In a recent study, an MIT research group showed substantial variability in link performance in terms of capacity and erasure rates (e.g., 10-50% erasure rates)in 802.11b mesh networks [2]. Gokhale *et al.* in [7] pointed out that the unpredictable behavior of the links seen in [2] was due to interference and not due to multi-path propagation effects as initially reported. Multi-hop ad-hoc networks used in defense or emergency response environments also exhibit such high residual erasure rates. These residual erasure rates (even after link layer error mitigation is done) have a substantial impact on end-to-end performance.

Gummadi *et al.* report that a range of selfish and malicious interferers can cause 802.11 performance to degrade much more significantly than expected[8]. Their experiments show that commodity 802.11 equipment is vulnerable to certain patterns of weak and narrow-band interference. Camp *et al.* performed an extensive measurement study on a multi-tier mesh network and showed that low-rate management and control packets can produce a dis-proportionally large degradation in data throughput and can lead to poor network utilization [4]. Sheth *et al.* look at the performance of WiFi-based long distance networks and characterize the packet loss [11]. For urban areas, they report high and variable raw/residual loss rates (retries and mac-acks were disabled) between 4-70%. The cause of these packet losses was found to be external WiFi interference i.e. other Wifi traffic on the same or adjacent channels. These results provide evidence that external interference is a significant source of packet losses in WiFi environments. Jameison and Balakrishnan note that even with a variety of PHY-level techniques, current systems rely heavily on link-layer retransmissions to recover from bit errors and achieve high capacity[9]. They also point out that designing an error-free communication link entails sacrificing significant capacity and that a design that allows some errors may be a better approach. Bianchi *et al.* report on the basis of measurements that 802.11g may experience severe inefficiencies when employed in an outdoor scenario [3].

Clearly, the measurement studies that have been performed so far indicate that high and variable packet loss rates are indeed possible with the primary cause being interference. Moreover, prior work in this area has not studied the issue of the three-way trade-off between exporting a low residual loss rate, low latency by minimizing ARQ retries and a high goodput. To our knowledge, this paper is the first to explore the problem of obtaining favorable trade-offs under high loss conditions and developing link and transport protocols that are suitable to such environments. We also look at the fine-grained behavior at the two layers and highlight some key interactions. Finally, we show how error-protection mechanisms need to be added to both the link and transport layers and provide protocols that balance the functionality of loss-tolerance across the two layers.

III. Experimental Measurements

In this section, we look at the experiment setup and discuss the results and insights. In the wireless driver, we make the ARQ (number of retransmission attempts per packet) user-settable. By default, this is 11. This makes it possible to study the impact of varying the ARQ on the link and transport performance. We also measure the number of data/control packets sent on the link and number of such transmissions lost in the face of noise. This gives us the raw loss rate at the link-layer. Finally, we measure the number of data packets sent by the transport layer to the link layer and the number of such packets that were not delivered even after the ARQ limit was reached. This gives us the residual loss rate exported by the link layer to the higher layers. At the transport layer (in the Linux kernel), we measure the number of packets and bytes of unique (new) data that were sent, number of packets and bytes that were retransmitted and the header overhead in each. We also measure he time instants when TCP timed out and the sequence number of the packet timing out, the Round-trip time (RTT) samples and the behavior of the smoothed RTT (SRTT) and the RTO. Finally, we study the behavior of the TCP congestion window. RTT samples are gathered with a granularity of 1 ms. The minimum RTO is 200ms, the maximum RTO is 12 seconds and the default RTO value is 3 seconds.

Similar to [11], we introduce controlled interference in the form of noise at the receiver to model varying amounts of interference from an external traffic source. We choose the 5.18 GHz (802.11a frequency range) so that we would not encounter interference from 802.11b wireless APs that are present in the rest of the building. The ORBIT setup does not have other wireless transmitters in this frequency which ensures RF isolation. While several runs of the experiments were conducted to ensure repeatability, a representative set of results is discussed in this paper. Moreover, the measurements were performed to study the trade-offs involved and get a trace model for a single wireless link which we use for single and multi-hop simulations on the ns-2 simulator. Hence, multi-hop experiments were not performed on the testbed.

Our main setup is to use three nodes: one acting as a wireless AP, one as a client and a node near the client acting as a sniffer. The wireless card on the sniffer is operated in a promiscuous mode and *tcpdump* is used to capture packets. By correlating the information gathered at the link layer, TCP layer and the *tcpdump* data traces, one can get an idea of the interactions happen between the link and transport layers. Figure 3 shows the topology. For the client, the number of ARQ attempts was set at 14 (for a total of 15 attempts per packet). To this setup, we add varying amounts

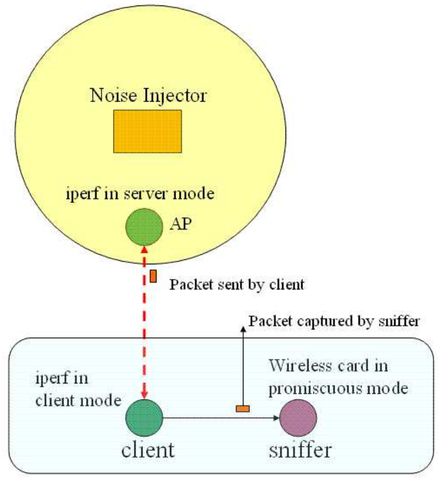


Fig. 3. Experimental Topology: The topology used to study the performance of the link and transport protocols and the interactions between them is as shown.

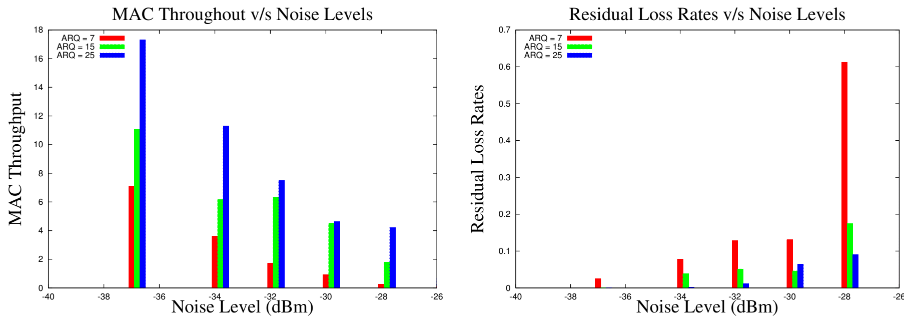
of noise through the Noise Injection Subsystem available on the testbed. The amount of noise injected was between -25dBm (a highly noisy environment) to -40dBm (an almost noise-free environment). Since we are mainly interested in the details of link and transport layer retransmissions and not in a complete reconstruction of the data flow, it is sufficient to capture packets at the single end of interest as shown.

A. Rate Adaptation and Goodput

Multi-rate retry is a technique used by 802.11 a/b/g wireless devices to make use of multi-rate capabilities in response to SNR degradation and packet corruption. Here, original and ARQ retransmissions of a packet can be sent at different data rates. We have disabled multi-rate retry in the driver so that a given packet is sent at the same rate for all of its transmission attempts. The wireless card however, may change its transmission rate for different packets based on the link quality it detects. For 802.11a, data packets may be sent at 54,48,36,24,18,12,9 or 6 Mb/s. We refer to the latter change of rates as rate adaptation.

Figures 1(a) and 1(b) show the performance impact as we go from an ARQ persistence of 7 to 15 to 25. As can be expected, the residual loss rate decreases with increasing ARQ. Also, the MAC level throughput increases since the link is utilized most of the time. However, an increase in link throughput does not guarantee an increase in link or transport goodput. At lower ARQs, transport-layer timeouts cause the link to be idle. The penalty paid however is an increase in the link latency (studied later). One of our goals is to provide high transport goodput and lower residual loss rate with low ARQ persistence.

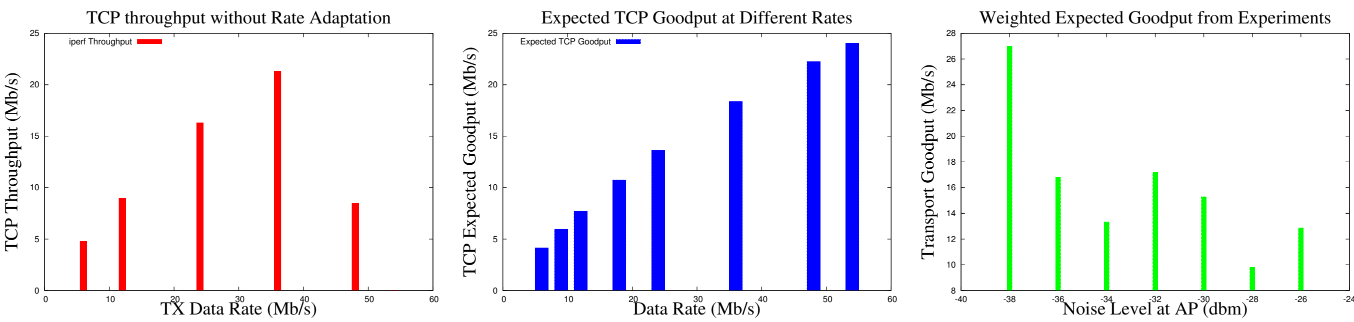
Figure 2(a) shows the performance when rate adaptation is turned off and the wireless card operates only at the data rate shown when the Noise Injector is turned off. However, the ambient noise level is high enough that transmission at 54Mb/s suffers. Rate adaptation is useful under conditions of noise and propagation errors and can be detrimental in case of interference-related errors. Here we see that turning off rate adaptation can be harmful since the cause of experimental



(a) MAC Throughput

(b) Residual Loss Rates

Fig. 1. We see the MAC-level throughput and the residual loss rates for three different ARQ persistence levels for a number of interference-scenarios. As expected, the residual loss rate is lowered as the ARQ persistence increases. The penalty paid is an increase in latency and lowered goodput.

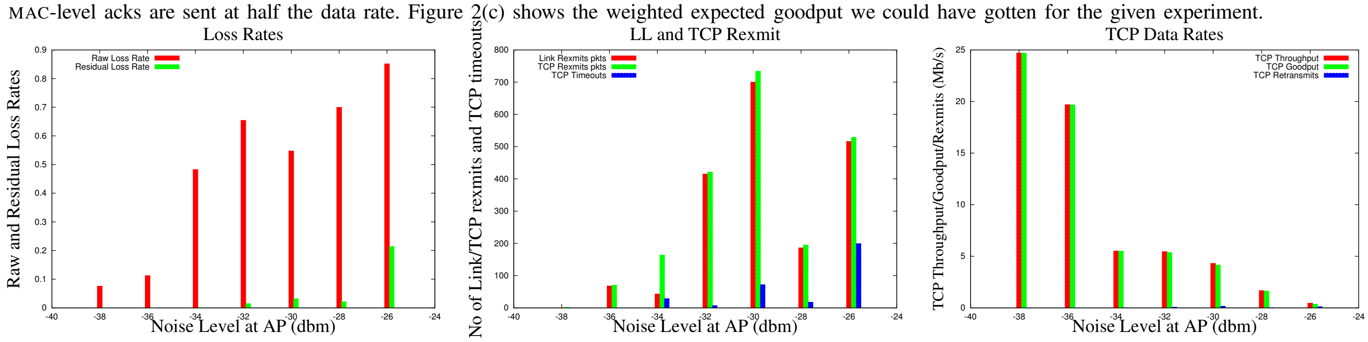


(a) Baseline: No Rate adaptation

(b) Expected Goodput (no loss)

(c) Expected Goodput (expt.)

Fig. 2. In Figure 2(a), without Rate Adaptation, the throughput obtained is maximum at 36 Mb/s setting at the baseline setting (no noise). Figure 2(b) shows the maximum expected goodput at the transport layer for different transmission data rates. TCP data packets and acks are sent at the data rate (X axis). The MAC-level acks are sent at half the data rate. Figure 2(c) shows the weighted expected goodput we could have gotten for the given experiment.



(a) Loss Rates

(b) Retransmissions and Timeouts

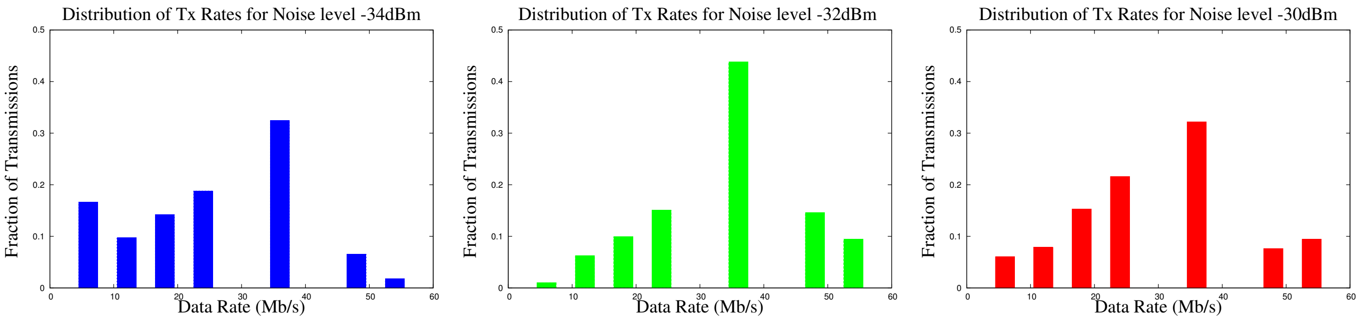
(c) Statistics of TCP performance

Fig. 4. For noise level -30dBm, Figure 4(a) shows the raw loss rate at the link layer and the residual loss rate exported to the upper layer. We see that the raw loss rate can be quite high which leads to a significant residual loss rate seen by TCP even with a high ARQ limit. Figure 4(b) shows the number of retransmitted packets at the two layers and the number of TCP timeouts. Figure 4(c) shows the transport-layer throughput, goodput and retransmitted bytes. It should be noted that as the noise level increases, the performance falls sharply.

errors is noise and not interference even though we model interference using noise. For this reason, we turn on rate adaptation for the experiments.

Figure 2(b) shows the theoretically *expected* goodput at the transport layer under conditions of no loss for different transmission rates. The MAC-layer acks are typically sent at half the transmission data rate selected by the driver. For example, if the data packets are sent at 48 Mb/s, MAC-acks are sent at 24 Mb/s. The values shown in Figure 2(b) are calculated by taking into account four different transmissions that happen per TCP packet sent, namely: the TCP data packet from sender to receiver, the MAC-layer ack for the same, the TCP ack from the receiver to sender and the MAC-level ack for the same. We also take into account the SIFS and DIFS intervals between transmissions, the PHY layer synchronization preamble time, the number of bits per symbol (this depends on the data rate), the symbol time and the packet size including TCP, IP and SNAP headers.

Figures 5(a), 5(b) and 5(c) show the distribution of transmission data rates for three different noise levels. From this set of measurements, we compute the expected TCP goodput we should see in each experiment. For example, at a fixed

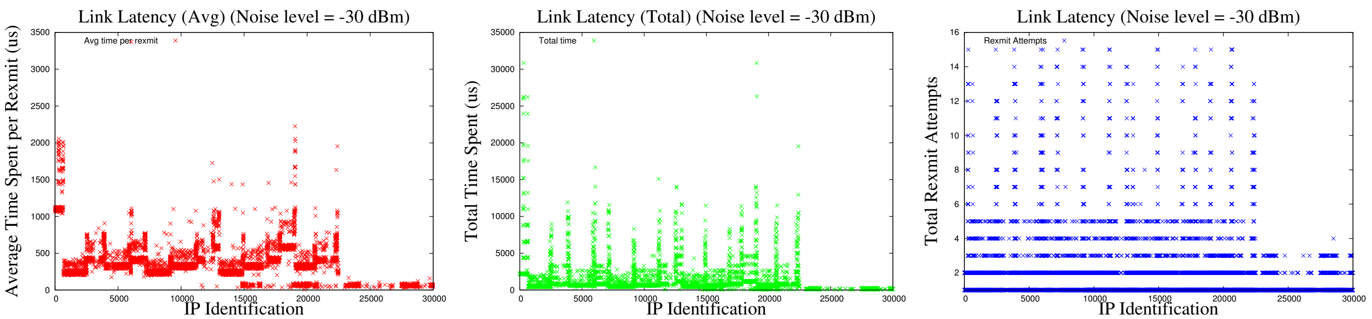


(a) Noise level = -34 dBm

(b) Noise level = -32 dBm

(c) Noise level = -30 dBm

Fig. 5. This figure shows the distribution of transmission rates for three different interference-levels. From this distribution and from the expected TCP goodput at each data rate, we can compute a weighted average of the expected TCP goodput for each experiment scenario.



(a) Avg Time per ARQ attempt

(b) Total Latency

(c) Number of ARQ attempts taken

Fig. 6. This figure shows the latencies incurred on the link layer for noise level -30 dBm. Figure 6(a) shows the average link latency per retransmission attempt in microseconds. Note that there is no back-off between successive retransmissions. Figure 6(b) shows the total time spent for a given packet. Figure 6(c) shows the number of retransmission attempts make for different packets. The X axis shows the IP identifier for a packet.

noise level of -30 dBm (seen in Figure 5(c)), around 50 % of the transmissions were made at data rates of 24 or 36 Mb/s. By taking the weighted average of the different transport level goodputs expected at the different data rates (shown in Figure 2(c)) and then factoring in the raw loss rate seen on the link, we estimate what the best-case TCP layer goodput we can get. The difference between what we get from our measured results and approximate expected goodput is what is lost as a result of protocol inefficiencies. For example, at the noise level of -30 dBm, the expected transport goodput under no-loss conditions based on the weighted average was 15.3 Mb/s. From the experiments, we see that the raw loss rate was 54 %, which means we can ideally get $15.3 \times (1.0 - 0.54) = 6.89 Mb/s$. We see that we instead get only 4.81 Mb/s, and the remaining was lost due to protocol inefficiencies.

B. *ARQ and Latency Impacts*

Figure 6 shows the impact of ARQ and retransmissions on the latency for noise level -30dBm. As noted in [4] and seen in our traces, the wireless driver will not back-off (exponentially) if it does not detect another packet in the air. Thus the intervals between successive packet transmissions that we see are not exponentially backed-off. There is some randomization however. For example, for the packet with IP identifier 47613 under noise level -30dBm, the intervals between successive transmissions were 325, 432, 334, 549 microseconds and so on. With real interference (unlike interference modeled using noise), we would also see exponential back-offs between successive transmission attempts and the latency penalties would have been even more severe.

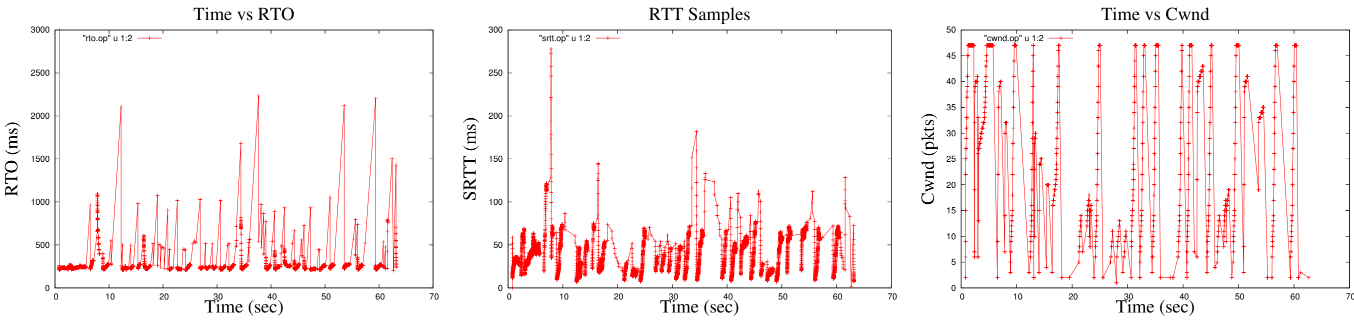
IV. Experimental Results

A. *Discussion of Loss Rates and TCP Performance*

Figure 4(a) shows the raw loss rate on the link at different noise levels. The figure also shows the residual loss rate that is exported to the transport layer. This residual loss rate is the loss rate that is seen by TCP subsequent to the 15 transmission attempts. It can be seen that the raw loss rate can rapidly increase and the performance (throughput and goodput) obtained degrades relatively quickly. Moreover, we can see that as the residual loss rate goes up beyond 5% (noise level -29dBm), the TCP performance drops rapidly (see Figure 4(c)). TCP is thus susceptible to even relatively low residual loss rates even under conditions of very small round trip times. This susceptibility will only increase as the round trip time increases.

B. *Discussion of Timeouts*

We now look at the instances of timeouts in our experiments (Figure 4(b)). As the noise level on the link increases, clearly the raw loss on the link goes up and consequently, the residual loss rate that is exported to the upper layer also goes up. While



(a) Retransmission Timeout

(b) Smoothed RTT

(c) Congestion Window

Fig. 7. Window and RTT plots (noise level -30dBm): These three figures show the behavior of the RTO, smoothed RTT (SRTT) and the congestion window in packets for noise level -30dBm.

we have a high number of ARQ attempts (1 original attempt + 14 retransmission attempts), there is evidence that some of the timeouts that occur are not simply due to the packet exceeding its link retransmission attempts but due to interactions with TCP. We now look at the instances of timeouts and categorize them based on their causes.

1) Timeouts due to loss: Most of the timeouts that we have observed are due to the fact that a packet that was given to the link layer by TCP could not be sent across the link because the ARQ limit of 15 was reached. In this case, if duplicate acknowledgments did not reach TCP (entire window was wiped out), then TCP is left with no option but to timeout and retransmit the packet. The vast majority of the more than 600 timeouts that we encountered were caused by this.

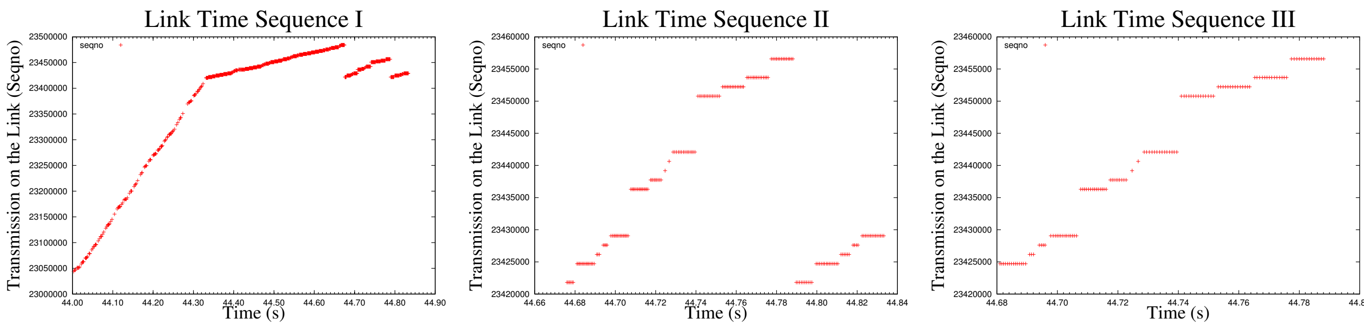
2) Timeouts due to latency and TCP interactions: Figures 7(a),7(b) and 7(c) show performance metrics of interest at TCP for noise level -30dBm. We see that large delay spikes can be encountered in the RTT samples leading to variable estimates of RTT. Moreover, the congestion window (to which the number of outstanding packet is close), can be high leading to high queuing delays at the link layers.

Packets can also suffer from increased packet latency on the link if packets that are being serviced before it incur a large number of ARQ attempts and thus high link delay. Figure 8(a) shows the number of attempts taken by intermediate packets between the 3 transmissions of TCP packet with relative seq no 23421817. The initial packet transmission failed and after 15 attempts, the link layer gives up. However, duplicate acks at the TCP layer detect the need to send this packet and the packet is retransmitted. However, this retransmitted packet is stuck behind a number of packets waiting to be delivered at the link layer several of which take almost 15 attempts (and some are delivered). The effect of this is that though the retransmitted packet is successfully delivered over the link in 7 attempts, TCP has timed out and the packet is then sent across the link a third time (this time taking 11 tries). Figures 8(b) and 8(c) also show the times taken by the intermediate packets. Note that since we have only 1 client, no exponential back-off is present. In the presence of a large number of nodes, exponential back-off would have exacerbated this effect. The retransmission timer RTO at TCP is 200 ms and is triggered by the large link latency seen by the second transmission attempt. This other cause of timeouts is harder to detect since it depends on protocol interactions between the link level driver and the transport protocol. Moreover,this effect manifests itself infrequently. It should also be noted that such interactions will be seen more frequently as the RTT increases (when we move from single hop to multi-hop scenarios) and as the bandwidth increases. Moreover, in the presence of multiple clients, exponential back-off between successive transmission attempts will exacerbate this effect.

In summary, we see that the performance of link and transport protocols degrades rapidly with increasing raw and residual loss rates. Moreover, this degradation will worsen with emerging scenarios such as multi-hop networks and metro-wide broadband which will present increased RTT and bandwidth. To obtain a good trade-off between link goodput, residual loss rate and link latency, it is important to design link protocols that can provide high goodput and low loss rate with low ARQ persistence. We present a link and transport protocol below which can provide an appropriate balance of the error-protection functionality across the two layers.

V. Protocol Design

Our link and transport layer schemes, while designed to operate independently, share some common features: adaptive granulation, estimation of loss rate statistics (mean, standard deviation), and FEC units organized pro-actively (PFEC, along with the original transmission) and reactively (RFEC, in response to feedback). We assume that the popular Reed-Solomon (RS) codes are used for FEC since they have powerful erasure correcting capabilities. As shown in Figure 9, each set of K data units (packets at TCP or fragments at link-layer) is protected with the addition of $N-K$ proactive FEC (PFEC) units to create a block of size N. The initial transmission attempt comprises these N units. The amount of PFEC to be added is determined by the estimated channel loss rate. Under a lossless scenario, no PFEC is added. If less than K units arrive uncorrupted at the receiver, RFEC units are sent to make up for the missing units. The amount of RFEC units to send is determined by the loss rate, amount of PFEC sent and the number of units still needed at the receiver



(a) Time Interval: 44.0-44.9 s

(b) Time Interval: 44.66-44.84 s

(c) Time Interval: 44.68-44.80 s

Fig. 8. For noise level -30 dBm, we see the occurrence of a spurious retransmission and timeout. The first figure shows the transmission of packets around the period of interest and we then zoom into this further. We see that the packet with seq no 23421817 is sent thrice. While attempt 2 is successful, the high latency incurred causes a timeout resulting in a third transmission of the packet.

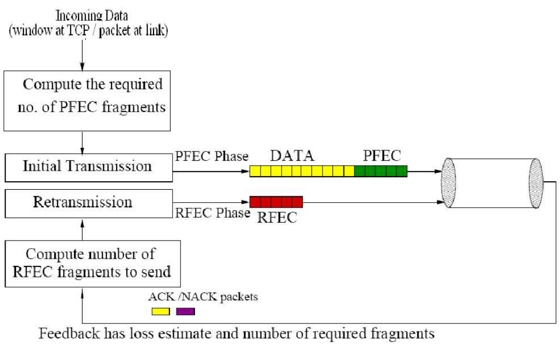


Fig. 9. Protocol Operation over an abstract lossy channel (could be either a single wireless lossy link or an end-end path experiencing loss): The initial data+PFEC transmission leads to feedback that determines the amount of RFEC sent to recover that block.

to recover the frame. Due to the sequence-agnostic property of FEC, the receiver needs to receive *any* K units (of data, PFEC or RFEC) to reconstruct the original K data units (with RS codes). We describe the various building blocks below. A detailed description can be found in [12] and references within.

Loss Rate Estimation FEC overhead is set as a function of the short-term statistics of the loss process. We estimate the loss rate using an exponentially weighted moving average (EWMA) A of loss rate samples taken once per block.

$$A = \alpha * sample + (1 - \alpha) * A \qquad (1)$$

The α parameter for the link and transport layers in our protocol designs are chosen empirically to be 0.005 and 0.5 respectively. A block refers to the fragments of a single packet at the link layer; hence a small value of α translates to averaging over several packets, as the samples are much more frequent. Since a block at the transport layer is composed of multiple packets, a higher value of α is used, as samples of loss are less frequent and we weigh the latest sample more. Our protocols are relatively insensitive to the value of α chosen.

Adaptive Granulation: A window of data at the transport layer and a single frame at the link layer is chosen to be the data part of a FEC block. The block (data + PFEC units) is fragmented into units which are subject to potential erasure.

10 Flows, Single-link	ERROR RATE					
PARAMETER	0 %	10 %	20 %	30 %	40 %	50 %
Goodput (Mb/s)	9.96	8.05	6.71	5.61	4.58	3.59
Throughput (Mb/s)	9.99	9.98	9.99	9.99	9.98	9.99
Residual Loss Rate (%)	0.00	0.00	0.00	0.00	0.00	0.00
Avg. Latency (ms)	10.97	13.83	13.26	13.80	14.67	15.05
PFEC Sent (Mb/s)	0.02	1.48	2.90	3.77	4.60	5.40
RFEC Sent (Mb/s)	0.00	0.39	0.36	0.59	0.80	0.98
PFEC Wasted (Mb/s)	0.02	0.61	1.04	1.03	1.02	1.00
RFEC Wasted (Mb/s)	0.00	0.26	0.23	0.33	0.38	0.39

TABLE I

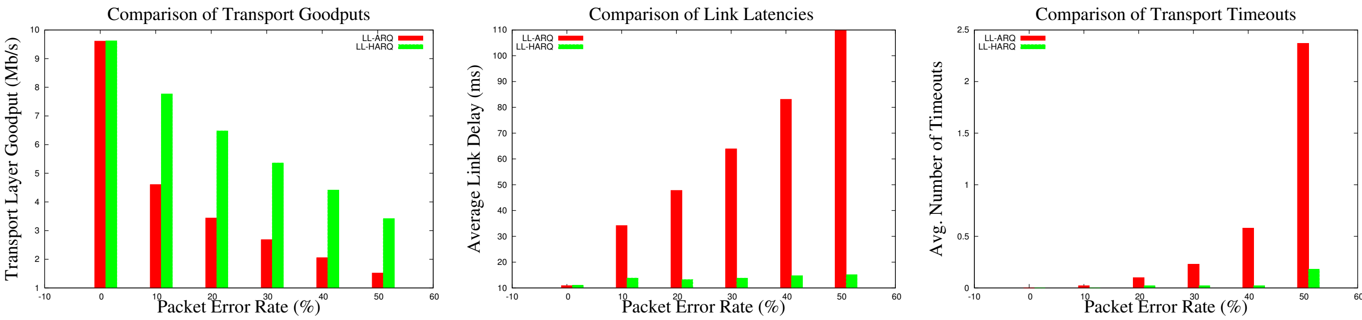
LL-HARQ DETAILS: THE VARIOUS PERFORMANCE METRICS FOR THE LL-HARQ SCHEME ARE SHOWN. THE RESIDUAL LOSS RATE EXPORTED TO THE TRANSPORT LAYER IN ALL CASES IS 0. ALSO, COMPARED TO THE AMOUNT SENT, THE WASTAGE IN PFEC AND RFEC IS SMALL.

For example, at the link-layer, a frame is fragmented into 20 units. The size of each unit is finalized once the amount of proactive FEC (PFEC) units per-frame is determined (see below). At the transport layer, the size of the current window determines the size of the block. Also, the FEC is sent from within *cwnd*. The segments are sized based on a desired minimum granulation (10 segments), subject to constraints on segment size.

Proactive 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 or outage.

Reactive FEC: 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 is computed based on the number of PFEC units already sent, number of units still needed to decode the data and the current estimate of the loss rate (see [12] for details).

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. At the transport layer, TCP acknowledg-

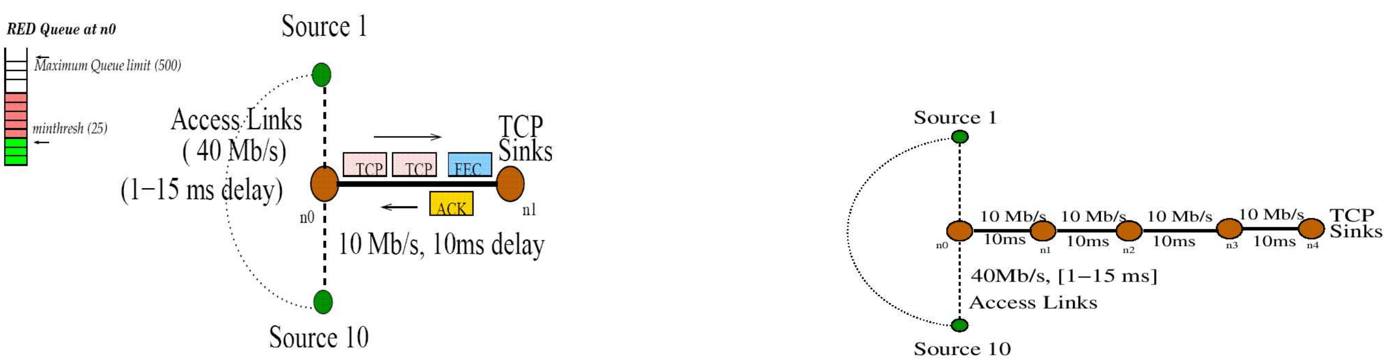


(a) Transport Goodput

(b) Link Delay

(c) Timeouts

Fig. 10. Synthetic Error Process (1 hop): We compare the transport layer goodput and the per-packet average link latency for the LL-ARQ and the LL-HARQ link protocols with LT-TCP as the transport protocol. It can be seen that the transport goodput and the latency are much better with LL-HARQ. Both variants export zero residual loss rate to TCP but because LL-HARQ uses only 2 transmission attempts, the obtained trade-off is much better than with LL-ARQ.



(a) Single-hop Topology

(b) Multi-hop Topology

Fig. 11. Test Configuration: The figure shows the topologies used to test the developed protocols and compare them against baseline protocols. We use 1 hop and 4 hop topologies with abstract lossy links.

ments carry such feedback information (also present in SACK blocks).

VI. Performance Evaluation

This section tests the hypotheses underlying our design and demonstrates the efficacy of our proposed mechanisms with a set of trace-driven simulations. We also test the protocols using a synthetic error process to provide a more stressful scenario. The ns-2 simulator was used to evaluate the protocols. We also consider different combinations of the link and transport protocols to study the performance issues. We use both single lossy-bottleneck and multi-hop configurations with 10 flows with RED/ECN at the queue(s) (see Figures 11(a) and 11(b)). Each link is a 10 Mb/s bottleneck link with 10 ms one-way delay with erasure rates varying from 0% to 50% (uniform error rate). Routers and hosts are ECN-enabled with *minthresh* and *maxthresh* values are as shown. The simulations were run for 100 seconds, and results are averaged over a minimum of 6 randomized runs on the ns-2 simulator. Confidence intervals are shown where applicable.

LL-ARQ is the baseline link-level protocol which has a pure ARQ mechanism (limit on the number of ARQ attempts set to 15) but without FEC support similar to the link protocol we evaluated earlier. To be conservative, LL-ARQ does not incorporate the typical back-off mechanisms between ARQ retransmissions. **LL-HARQ** is our proposed link protocol which has a limit at most one ARQ retransmission attempt and includes PFEC and RFEC mechanisms as explained in section V. In both variants, an incoming packet at the link layer is split into 20 fragments. With LL-ARQ, retransmissions are the same as the original transmission. With LL-HARQ, retransmissions consist of RFEC fragments. LT-TCP and TCP-SACK are the transport layer protocols used.

We first test the protocols over the single and multi-hop topologies using the traces and then use the uniform error model with the packet error rate ranging from 0 to 50% for the synthetic error process. For the multi-hop simulations, we create identical links using the trace models we gathered for a single link from the measurements.

A. Evaluation of Protocols using Traces

The traces that were gathered were used to generate link error models which were then used as input for our ns-2 simulations. Figure 12 shows the cumulative distribution function (CDF) for the number of consecutive packet errors for different noise levels from our gathered traces. Recall that in our link layer protocol, each fragment of a packet is acted upon independently by the error process. We use the distribution shown in Figure 12 to model the fragment errors. We use the

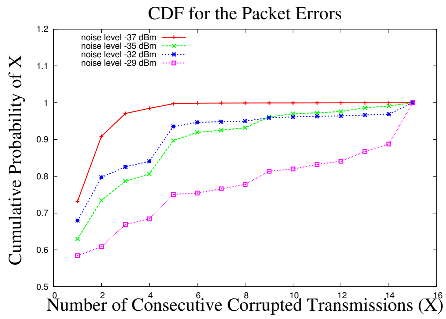


Fig. 12. The figure shows the cumulative distribution function for the number of consecutive packet errors caused by the injection of noise. We use this distribution to generate fragment errors for our ns-2 simulations.

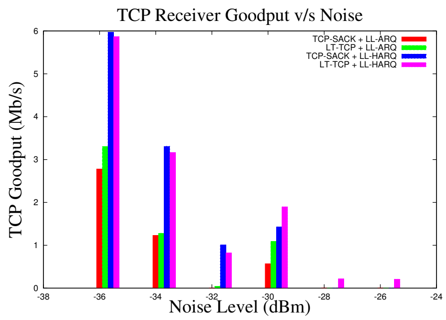


Fig. 14. Trace-Driven Error Process (4hops): This figure shows the performance of all four combinations of transport and link protocols over a 4 hop topology with 10 flows for a variety of interference levels. The errors were generated using the distribution generated from the traces. We see that under the worst case, LT-TCP support is needed at the transport layer to complement LL-HARQ at the link-layer.

1 and 4-hop topology shown in Figure 11(a) and 11(b).

Single hop topology: Figure 13(a) shows the transport layer (LT-TCP) goodput obtained with the two link layer protocols in use. We see that LT-TCP is able to improve the performance at the transport layer in spite of using only 1 ARQ retry. The significant gains by using LL-HARQ in lace of LL-ARQ can be seen in Figure 13(b). A single hop topology is used as shown in Figure 11(a).

Four-hop topology: We now evaluate the various protocol combinations using the traces. We use the four-hop scenario to see the transport layer goodput obtained with the various combinations (see Figure 14). We see that at lower noise levels, both TCP-SACK and LT-TCP give good performance (as long as LL-HARQ is present). LL-HARQ exports a very low residual loss rate and so both transport variants do well with LT-TCP doing marginally worse due to slight adaptive granulation overhead. However, as the noise level and loss rate increase, LT-TCP mechanisms kick in to provide better performance compared to TCP-SACK. Beyond noise-level -30dBm, only the combination of LT-TCP + LL-HARQ is able to provide any performance.

B. Link Protocol (LL-HARQ) Evaluation with Synthetic Traces

Single-hop topology: Figure 10(a) shows the transport layer goodput obtained with LL-ARQ and LL-HARQ with TCP-SACK as the transport protocol (results with LT-TCP are similar). We note that both LL-ARQ and LL-HARQ export a negligible loss rate to the transport layer. However, LL-HARQ

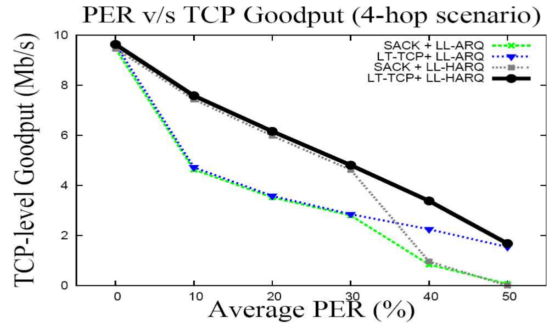


Fig. 15. Synthetic Error Process (4 hops): The transport-layer goodput for the 4-hop scenario is shown in this graph. With TCP-SACK as the transport protocol, the performance collapses beyond 30%. With LT-TCP however, the degradation in performance is linear, especially with LL-HARQ as the link protocol. LL-HARQ also leads to lower link latencies compared to LL-ARQ.

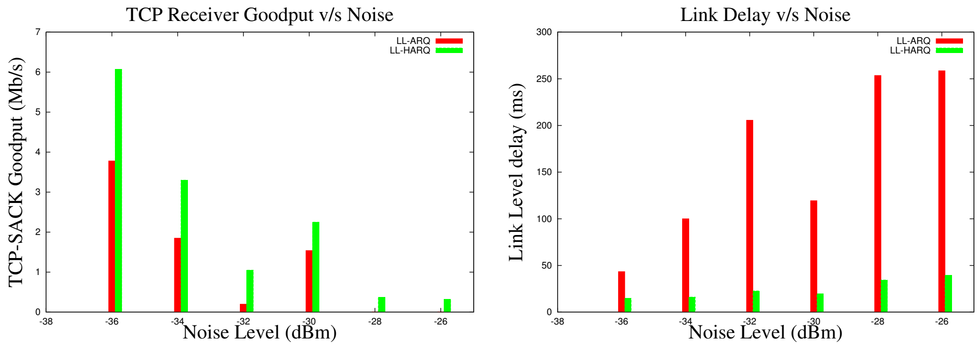
manages to do this with just two transmission attempts (1 ARQ) which keeps the link latency low. LL-ARQ relies on a high number of ARQ attempts (15) to achieve a low residual loss rate causing high link latency (see Figure 10(b)). Figure 10(c) compares the average number of timeouts incurred at the TCP layer. Table I shows the detailed performance metrics for LL-HARQ. From the table we see that we can get good performance even at high loss rates. For example, even with a packet error rate of 50%, we are able to get a transport goodput of 3.59 Mb/s with just 1 ARQ. The residual loss rate is zero and using a single ARQ attempt limits the average latency to around 15 ms.

Four-hop topology: Figure 15 shows the performance of the four protocol combinations using 10 flows on the 4-hop topology and a *bursty* error model where the actual error rate is varied around the nominal error rate. As can be seen, LL-HARQ is able to provide significant performance benefits till an average error rate of 30 %. However,under the worst-case conditions of high and bursty errors and multiple hops, LL-HARQ support is insufficient and TCP-SACK performance collapses and transport support in the form of LT-TCP is needed..We thus find that the improvements by the transport (LT-TCP) protocol to goodput and link (LL-HARQ) protocol to latency complement each other. Thus, these modifications help achieve our key objectives of low residual loss rate on the link, low average latency (average number of transmission attempts is less than 2), high link and transport-level goodput.

In summary, multi-hop scenarios require the combined LT-TCP+LL-HARQ scheme to deliver good performance (low link latency, low link residual loss rate, high link and transport goodput) especially at high link-layer loss rates. Use of adaptive HARQ/FEC at both link and transport layers enables applications to tolerate extremely bursty and persistent loss conditions up to the tune of 50% average PER and still achieve high transport-layer goodput and low latency.

VII. Conclusion

Recent wireless measurement studies have provided evidence of significant performance degradation of wireless links and networks in the face of interference. In this paper, we looked at the current approaches at the link and transport layers under an interference regime modeled through the injection



(a) Transport Goodput

(b) Link Latencies

Fig. 13. Trace-Driven Error Process (1hop): Transport Layer goodput and link latencies using LL-ARQ and LL-HARQ for a single hop topology. We see that similar to the synthetic error model results, LL-HARQ gives much better performance over LL-ARQ in terms of link goodput (leading to better transport goodput), link latency and residual loss rate.

of noise. We performed a detailed study of the trade-offs inherent in a link design and saw how having a large number of ARQ retransmissions reduces the residual loss rate exported to the upper layers at the expense of increased latency and decreased goodput. To the best of our knowledge, a structured approach that looks at the three-way trade-off between loss rate, latency and goodput and presents protocol solutions has not been pursued prior to this. To investigate current trade-offs, experiments were performed on the ORBIT wireless testbed at Rutgers University.

We saw that the performance of the link and transport protocols are affected by varying levels of interference. High ARQ persistence can help bring the raw loss rate down at the link layer which is exported as small residual loss rate to the TCP layer. However, the high ARQ persistence extracts a cost in terms of high link latency which leads to high and variable delays. Moreover, such delay spikes can lead to subtle interactions between the link and transport layers such as spurious retransmissions and timeouts. At the link layer, the high raw loss rate leads to reduced throughput and goodput. At the transport layer, a residual error rate of around 5 % causes the connection to collapse.

To achieve a favorable trade-off between goodput, residual loss rate and latency, we have designed (in prior work) a link protocol that can deliver high goodput and low residual loss rate with just 1 ARQ retransmission. However, at high loss rates and over multiple hops, residual losses on each hop can aggregate to result in significant end-to-end loss rates. To solve such residual issues and to complement LL-HARQ, a transport protocol designed along similar lines called Loss-Tolerant TCP (LT-TCP) was developed. We compared the baseline LL-ARQ and proposed protocol LL-HARQ using trace-driven (from the data traces collected) as well as synthetic error models on the ns-2 simulator. We saw how the three-way trade-off can be improved dramatically with LL-HARQ as compared to LL-ARQ. Similarly, we showed the utility of LT-TCP over standard transport protocols such as TCP-SACK. We also provided insights into the *balance of functionality* between the link and transport layers for error-protection. While LT-TCP and LL-HARQ are designed to be deployed independently, synergistic benefits can be obtained by operating them in conjunction.

References

[1] Open-Access Research Testbed for Next-Generation Wireless Networks (ORBIT). URL: http://www.orbit-lab.org.

[2] 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.

[3] G. Bianchi, F. Formisano, and D. Giustiniano. 802.11b/g Link Level Measurements for an Outdoor Wireless Campus Network. In *WOWMOM '06: Proceedings of the 2006 International Symposium on on World of Wireless, Mobile and Multimedia Networks*, pages 525–530, Washington, DC, USA, 2006. IEEE Computer Society.

[4] J. Camp, V. Mancuso, O. Gurewitz, and E. W. Knightly. A Measurement Study of Multiplicative Overhead Effects in Wireless Networks. In *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pages 76–80, Phoenix, AZ, USA, April 2008.

[5] K. Chebrolu, B. Raman, and S. Sen. Long-distance 802.11b Links: Performance Measurements and Experience. In *MOBICOM*, pages 74–85, 2006.

[6] S. Dawkins, G. Montenegro, M. Kojo, V. Magret, and N. Vaidya. End-to-end Performance Implications of Links with Errors. IETF RFC 3155, August 2001.

[7] D. Gokhale, S. Sen, K. Chebrolu, and B. Raman. On the feasibility of the link abstraction in (rural) mesh networks. In *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pages 61–65, Phoenix, AZ, USA, April 2008.

[8] R. Gummadi, D. Wetherall, B. Greenstein, and S. Seshan. Understanding and Mitigating the Impact of RF Interference on 802.11 Networks. *SIGCOMM Comput. Commun. Rev.*, 37(4):385–396, 2007.

[9] K. Jamieson and H. Balakrishnan. PPR: Partial Packet Recovery for Wireless Networks. *SIGCOMM Comput. Commun. Rev.*, 37(4):409–420, 2007.

[10] Reiner Ludwig and Randy H. Katz. The Eifel Algorithm: Making TCP Robust Against Spurious Retransmissions. *SIGCOMM Comput. Commun. Rev.*, 30(1):30–36, 2000.

[11] A. Sheth, S. Nedevschi, R. Patra, S. Surana, E. Brewer, and L. Subramanian. Packet Loss Characterization in WiFi-Based Long Distance Networks. In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pages 312–320, Anchorage, AK, May 2007.

[12] V. Subramanian, S. Kalyanaraman, and K.K.Ramakrishnan. Balancing Loss-Tolerance between Link and Transport Layers in Multi-Hop Wireless Networks. In *The 15th IEEE Workshop on Local and Metropolitan Area Networks*, Princeton, New Jersey, USA, June 2007.