Analysis of XCP in a Wireless Environment

George Nychis, Gaurang Sardesai, and Srinivasan Seshan
Carnegie Mellon University
{gnychis,gsardesa,srini}@cmu.edu

***Abstract*— Previous works on TCP have shown instability and inefficiency when there is non-congestion loss present in the network. The eXplicit Control Protocol was designed to increase stability and efficiency of high bandwidth-delay product networks. To do this XCP has decoupled its fairness and efficiency controller and provides explicit feedback.**

We do a comparison of the two transport layer protocols to find out if XCP performs better than TCP with non-congestion related loss. We examine several properties of the eXplicit Control Protocol which we believe can improve throughput in lossy wireless networks. We also examine shortcoming of the XCP protocol which are brought into focus over wireless networks, causing instability and inefficiency. We show that while XCP is not perfect in a shared medium, it does perform better than TCP, and in certain cases, achieves 200% throughput over TCP.

I. Introduction

THE lack of the TCP's ability to distinguish between congestion related loss and non-congestion loss leads to inefficiency in the protocol. This is especially apparent in high bandwidth-delay product networks, like wireless networks. The eXplicit Control Protocol was developed by Katabi et al. [1] and has several properties which we show through our analysis to provide higher throughput in lossy wireless networks. While higher throughput in a lossy 802.11 wireless network is shown to be achievable through XCP, we will also theoretically explore other properties of XCP such as fairness and leave the analysis and improvements to XCP for future work.

Through our analysis we will show that XCP can achieve 200% throughput over TCP in certain loss scenarios, and has increased efficiency with the number of hops over TCP in multi-hop scenarios. We will also show how a shared medium effects XCP's ability to converge to efficiency and how the inability to predict capacity in wireless networks creates oscillations and improper feedback. This has also been shown by Zhang and Henderson [2] in their study of XCP in wireless networks, but never before has anyone analyzed XCP in a true wireless network with the 802.11 MAC protocol.

The rest of the paper is organized as follows. Section 2 deals with related work in this area. In section 3, we investigate the theoretical framework of XCP and explore why we expect it to perform better than TCP. Section 4 outlines our experimental setup and our testing methodology. We analyze our results in section 5, talk a bit about our speculations and future work in section 6 and conclude our findings in section 7.

II. Related Work

A. XCP Experimental Studies

There have been two experimental studies [2], [3] of the eXplicit Control Protocol since the original Katabi et al. paper [1]. Zhang and Henderson's work through their Linux implementation has been the only other work to evaluate XCP's performance within a wireless network [2]. Their evaluation only looked at a last-hop wireless infrastructure network and was tested within a completely wired environment with probabilistic loss. The loss was injected into one of three locations: before the wireless hop, during the wireless transmission from base station to receiver, and anywhere in the return direction. The loss probability was varied between .0001 and .1 to cover a wide range of loss ratios typical of wireless networks. Along with analysis on different loss rates and location of the loss, the authors analyzed XCP performance with and without the SACK option against a TCP with SACK enabled case. Their evaluation shows XCP handles non congestion loss better than TCP in terms of bandwidth utilization in all cases, but XCP bandwidth utilization quickly degrades with losses even at a low rate of 0.001 without SACK and at 0.025 with SACK. One limitation of this study of XCP in a wireless network, when loss is created artificially, is the lack of interaction with rate adaptation. The design of our evaluation was done with this in mind so as to create actual losses, capturing the effects of rate adaption in the analysis.

Mosebekk's linux implementation and analysis of XCP was conducted to show the benefits of XCP in a wired environment [3]. In Mosebekk's evaluation within a wired testbed, the results show that XCP matches its simulated results shown in [1]. Both XCP experimental studies show that XCP fails to converge under circumstances such as clients not using their allowed bandwidth, inaccurate router settings and rate adaptive networks which create estimation problems in the feedback control. Mosebekk's implementation of XCP did not include implementation of XCP's interaction with loss in a network and therefore could not be used for our evaluation.

B. TCP Analysis and Improvements with Loss

There has also been a large number of studies of TCP in lossy wireless networks due to TCP's poor performance in networks with non-congestion related loss [4]–[8]. Balakrishnan et al. propose the snoop protocol [4] to enhance TCP's performance in lossy infrastructure wireless networks by caching unacknowledged TCP data at the base station and locally retransmitting it based on acknowledgment and time-out policies. The snoop protocol improves TCP performance without modifications to TCP and its end-to-end semantics. In the evaluation of the snoop protocol, the authors show improvement of TCP's throughput of up to 20 times over regular TCP.

Other studies of TCP in wireless have looked at its performance in multi-hop ad hoc networks, such as Dyer and

Boppana's study of TCP in three ad hoc routing protocols: AODV, ADV, and DSR [6]. In Dyer and Boppana's study they show that TCP over the three protocols can benefit from a new technique they introduced, fixed-RTO, as well as SACK and delayed ACKs. Their fixed-RTO technique aims to distinguish between route loss and network congestion by identifying route loss as consecutive timeouts from a missing ACK not being received before the second RTO timeout expires. Through their analysis fixed-RTO improves DSR's throughput up to 75%, but only provides a marginal increase for AODV and ADV.

Holland and Vaidya propose an explicit link failure notification (ELFN) [5] to improve TCPs performance in response to explicit congestion notifications. They disable congestion control mechanisms until the link is back up again, and use probe packets to determine the link's availability. They compare performance by varying probe packet intervals, RTO and congestion window sizes, and the choice of probe packets. Their work of ELFN focused on mobile ad hoc networks using DSR as the routing protocol and was built off of earlier works on TCP and explicit loss notifications such as Floyd's explicit congestion notification (ECN) [9]–[12].

Lim et al propose a scheme called backup routing [8] which uses multipath routing to ensure that routing can occur in the face of frequent link failures. They use the Split Multipath Routing (SMR) protocol over DSR, and claim a 30% improvement. Klemm et al take a different approach and suggest using signal strength based link measurement [7] where they either change transmission power reactively, or discover new routes proactively. They use AODV over ad hoc networks and claim 14-30% improvement in goodput.

III. Observations in XCP for Wireless

The major area in which we believe XCP can improve over TCP in wireless networks is TCP's period of inefficiency after encountering a loss. TCP reacts to loss by cutting its window size in half and additively increasing it to a point where TCP thinks the network is full utilized. This behavior is often shown as a sawtooth, sharply cutting down on loss, and slowly ramping the connection rate back up. As Katabi et al [1] state, it may take thousands of RTT's for TCP to reach full utilization.

When experiencing non-congestion related losses, TCP will still react to the loss as if it were congestion. The congestion control algorithm will cut the sender window size in half, and then begin to increase it additively thereafter in the congestion avoidance stage. This behavior can lead to major inefficiency as TCP additively increases its window and possibly wastes thousands of RTT's slowly converging to the maximum utilization. With bursty losses, common in wireless networks, TCP may even time out. The greater the loss rate, the greater the number of cuts in the window, and the longer it will take TCP to converge to maximum utilization.

This behavior is plotted in figure 1 from a 5 second throughput test with 1% packet loss. TCP is shown to reach what we have determined to be the links full capacity with the initial slow start stage, but thereafter it is only able to achieve 2/3 of the links capacity at maximum, and has numerous lengthy congestion avoidance stages where the link is underutilized. We will further show how XCP can achieve shorter inefficient stages after loss in theory to increase efficiency.

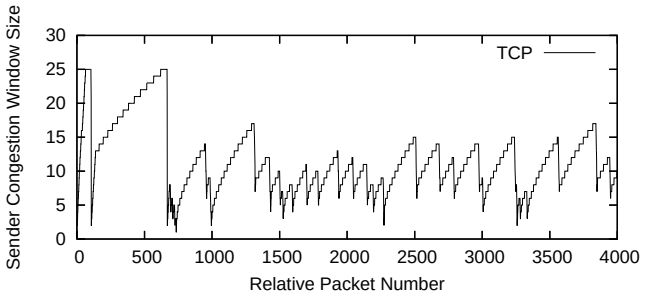

Fig. 1. TCP's reaction to a 5 second period of 1% packet loss. We make the observation that TCP is inefficient during the additive increase stages after congestion, and XCP would only be in this additive stage for a single RTT, increasing efficiency. The large peak at the beginning of the transmission was the highest TCP achieved, and packet loss there after prevented it from ever reaching this efficiency level.

Where XCP can improve over TCP is its smaller period of inefficiency after loss. We believe XCP has three important properties which allow for greater efficiency in lossy networks. These three properties of XCP's protocol are: the explicit feedback from the network, the efficiency controller, and the feedback filter.

The first two important properties of the eXplicit Control Protocol is its efficiency controller and congestion header. The efficiency controller in XCP is designed to keep queue size to a minimum, reduce loss due to over sized queues, and keep the link utilization to a maximum. The important observation here is that XCP will compute the increase or decrease in the sender's window based on several parameters, one of which is the currently available bandwidth. By making the feedback proportional to the available bandwidth, XCP can quickly adapt to changes in the environment. This would allow quick convergence to full efficiency whereas TCP exponentially converges to efficiency during slow start, but additively elsewhere. This is made possible through the congestion header which has a field for the sender's current rate and a field for the network to provide this feedback by increasing or decreasing window size.

XCP's ability to increase or decrease the sender's window explicitly would allow XCP to decrease the duration of inefficiency after loss. For example, after loss occurs and the sender reduces its rate, this new rate will be seen in the network through the congestion header of the next successful packet. Given no other flows in the network, once the sender reduces its rate the router's available bandwidth will increase. When the routers inspect the next successful header and observe the reduced rate along with their increase in available bandwidth, the routers will provide positive feedback proportional to the available bandwidth. This increases the rate of the sender back up to full efficiency without long durations of inefficiency while probing the network for spare bandwidth as TCP does.

The final important property of XCP which defines the exact duration of the inefficiency period after loss is the feedback filter. A feedback filter is important in any explicit congestion

control mechanism to prevent oscillation, instability, and inefficiency. This is noted by Jain et al [14] in their implementation of the DECbit, stating that the frequency at which feedback is given by the network to the users can lead to oscillations if too high or slow adaptation if too low. The authors, relating to control theory, determine the optimal feedback frequency to be the time from when any given feedback is applied to when its effects can be observed by the controller in the network. Both XCP and the DECbit scheme determine the feedback frequency to be a single RTT since it is the time it takes the feedback to reach the user plus the time the router can observe the effects. Therefore, the exact duration of the inefficiency period of XCP is one RTT, as opposed to the possibility of thousands of RTT's for TCP.

For the given reasons, XCP should perform better with non-congestion related losses in environments such as wireless networks. To support our beliefs we have done experimental tests in a real wireless testbed to show XCP's interaction with non-congestion related loss. We will explore our results and show how they support our beliefs in the following sections.

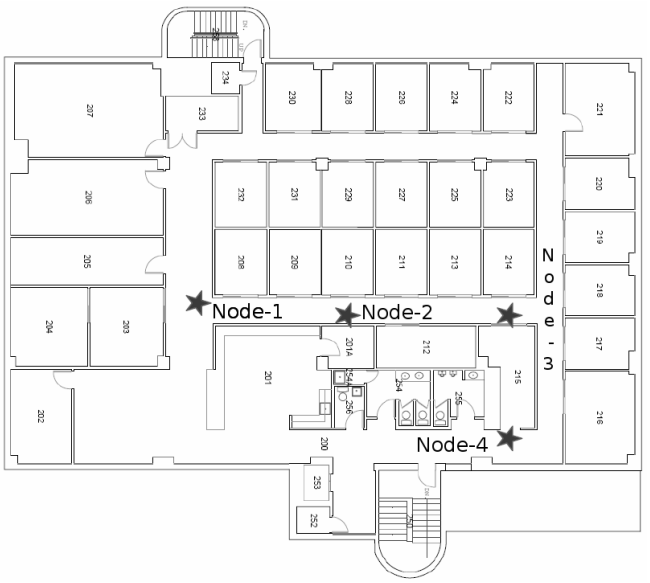

Fig. 2. This floor map shows where our nodes were setup within our department.

IV. Experimental Setup

To compare the performance of TCP and XCP in a lossy environment we setup a testbed in our department, illustrated in figure 2, and generated artificial loss. Our experiment allows XCP to interact with the 802.11 MAC protocol, contention in a shared medium, and bursty loss, which the wired simulation in Zhang and Henderson's wireless analysis could not capture. We compare throughput for the two protocols, however we make several observations that will lead to comparison with other metrics in future work. The topology and setup of our experiment follows closely with the experiments run by Jinyang et al [13] which measured the capacity of wireless networks. The topology for our analysis section was a chain topology and is shown in the figure. CTS/RTS was enabled for all transmissions due to the hidden terminal problems in our topology.

Our experiments were conducted using 5 computers, running Zhang and Henderson's Linux 2.4.32 kernel XCP implementation, and MADWiFi drivers for Atheros chipset wireless cards. So as to not interfere with CMU's infrastructure 802.11b wireless network, while also ensuring it did not interfere with our experiments, we carried out all our tests using 802.11a.

Before any experiments were run we did 60 second link tests between each hop and ensured the capacity per hop was around 3000KB/sec. To generate random loss we used the Netem network emulator. Netem allowed us to generate loss at the kernel level based on a probability passed to it. We used the default loss model, which is the uniform distribution. Models such as normal and Pareto distribution can also be specified, but those can be implemented as future work. For future work, which will later be discussed, we would like to generate loss at the hardware level.

We adopt the loss model followed by Zhang and Henderson's wireless simulation for our experiments. To test the protocols at each loss probability, we systematically iterate through the probabilities running a 60 seconds throughput test with ttcp for both protocols. Zhang and Henderson's wireless simulation let XCP converge, as they claim it will obtain stable performance, however a bug in Netem has kept us from doing the same. We would like to question the importance of letting XCP converge first in our future work section. Unless otherwise stated, the capacity parameter was set to 54Mbit/sec. We will also discuss the sensitivity of this parameter in our analysis section.

We generated static routes between the hops by using iproute2. While it was possible to use Ad-Hoc routing algorithms like DSR, AODV etc, our intent was to test the performance of XCP over wireless independent of any routing protocol. Hence we opted to forgo routing algorithms which might have introduced variability and chose the simplified static routing method instead.

To further smooth out our results, we ran 3 tests at each loss probability and averaged them, throwing away no results. This allowed us to observe the stability of both protocols and also account for interference or anomalies from our real world environment. We will now provide the results of our experiment.

V. Analysis

The analysis section will go through several experiments which we have run that show XCP can outperform TCP in terms of throughput. Experiments are done in direct client to server communication, and also in multi-hop scenarios. We also evaluate XCP's performance with SACK and without SACK enabled. We strove throughout to make sure that TCP and XCP had the same parameters for comparative runs, in terms of channel capacity and length of our runs. Since our environmental conditions changed so frequently, we ran comparative TCP and XCP runs consecutively, so that there was very little change in our conditions, like antenna displacement, and people walking.

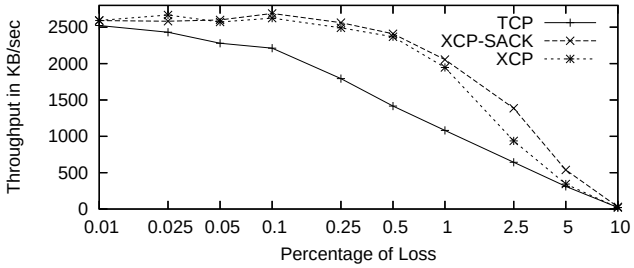

Fig. 3. TCP and XCP throughput with communication from Node-1 to Node-2. The graph also shows performance of XCP using SACK to XCP without SACK.

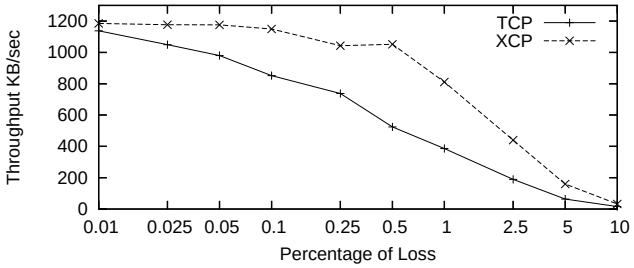

Fig. 4. TCP and XCP throughput with communication from Node-1 to Node-3. Routing was done statically at Node-2 and a .01% loss represents .01% loss per link.

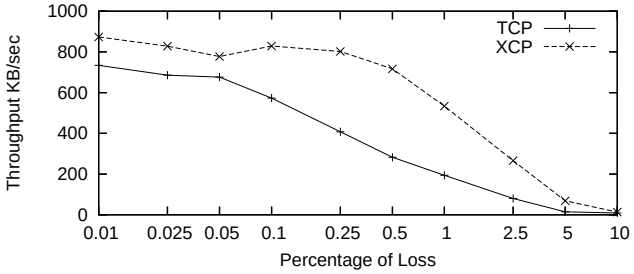

Fig. 5. TCP and XCP throughput with communication from Node-1 to Node-4. Routing was done statically at both Node-2 and Node-3. The graph shows that TCP consistently performs better than XCP in different scenarios, even with increased contention. XCP would also be able to scale to a larger number of hops than TCP.

A. TCP and XCP: Node-1 to Node-2

Our first experiment was run with a single flow between the first two nodes in the chain. Node-1 was the sending node and node-2 was the receiving node. We chose to run the experiment twice using XCP with SACK enabled and XCP with SACK disabled. This was done to also evaluate Zhang and Henderson's claims that SACK has major benefits for XCP's performance with non-congestion related loss. TCP was run with SACK enabled for all of our experiments. The results of this experiment are shown in figure 3.

The first observation is that XCP outperforms TCP in all loss probabilities. Figure 3 also shows that XCP can handle up to .1% loss before its throughput begins to degrade, where on the other hand TCP's throughput begins degrading with any amount of loss. We believe this supports our hypothesis that TCP becomes inefficient for greater periods of time than XCP when loss is introduced due to its additive increase. The results for XCP also support our hypothesis that XCP can respond quick enough to reduced rates within a single RTT to increase its efficiency back to 100%.

The gap in performance between the two protocols begins to increase at .1% and begins to converge back around 2.5%. The greatest performance difference is seen at 1% packet loss where XCP nearly doubles the throughput achieved by TCP. Intrigued by XCP's performance at this loss rate, we decided to tweak XCP's capacity parameter closer to the actual capacity we thought might be achievable. By limiting the maximum capacity around 24 Mbit/sec we were able to achieve an XCP throughput of 3013.51 KB/sec, almost triple the highest TCP reading we received of 1173.58 KB/sec.

Both protocols converge together at 10% loss. This was evident in all experiments we ran. Therefore even though XCP can perform better in terms of throughput in many different loss probabilities, neither protocol can exceed the 10% loss probability.

Although we had originally planned to run the experiments at loss rates beyond 10%, we gave up on that idea after noticing the abysmal rates both protocols provided us at high loss rates.

B. TCP and XCP: Multi-hop

In these next two experiments we setup a multi-hop scenario, using single flows still, between two sets of nodes, 1 and 3, also 1 and 4. For these experiments both XCP and TCP had SACK enabled. The absolute loss we tested at was applied to each link, such that for a loss of .01%, each link had a loss of .01%. The two figures which show our results are figures 4 and 5. These graphs confirm XCP's ability to achieve higher throughput than TCP in a multi-hop scenario also.

The graphs also show is that XCP consistently performs better than TCP, not only in point to point communication such as our first experiment. XCP's ability to perform better than TCP in the multi-hop scenario would also allow greater scalability of reliable end to end communication in a multi-hop environment with loss.

XCP also becomes more robust as the number of hops increase, figure 5 shows the performance gap increasing between XCP and TCP in the loss range of 0.1 to 1. We speculate that as the number of hops increase, the performance increase between XCP and TCP will grow in favor of XCP.

C. Stability of XCP in Wireless

In this section we would like to analyze XCP's stability in wireless. Katabi et al, Mossebek, and Zhang et al have all shown XCP to be stable in full duplex wired networks, however our results show that XCP's performance is extremely variable in wireless networks, even with 0% packet loss. This follows closely to Zhang and Henderson's analysis of XCP in contention based mediums, showing utilization to oscillate over a period of 60 seconds. Figure 6 shows the high, low, and average for the three 60 second throughput tests we ran at each probability. These results show that between our three different runs we experienced a high level of variance.

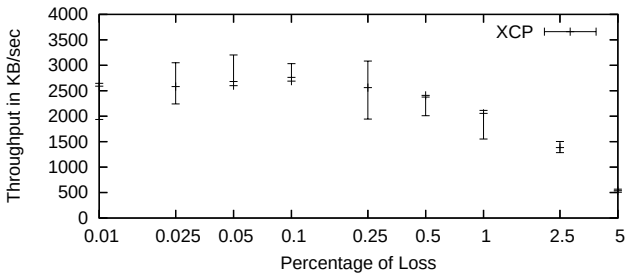

Fig. 6. The high, low, and average for the three 60 second throughput tests run for each probability. The graph shows XCP only, however we experienced similar results for TCP, possibly due to the properties of wireless.

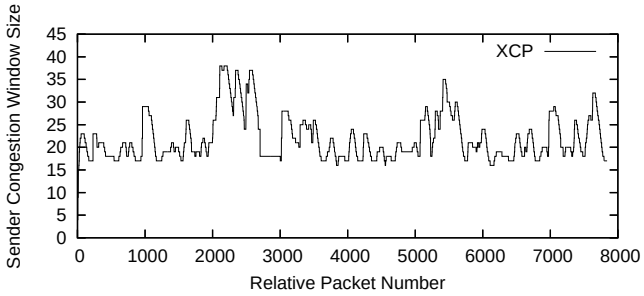

Fig. 7. This behavior of the sender congestion window size was observed in 7 of our 10 stability analysis experiments. It shows an amount of oscillation in XCP's utilization of the link from what we believe is due to the wireless shared medium and contention.

To further study this phenomenon we added printk()'s to the kernel, printing out the sender congestion window size before every packet was sent. The capacity was set to 24 Mbit/sec for this experiment and no loss was introduced to the link. We chose 24 Mbit/sec, trying to match the highest throughput we had seen achievable. We ran the experiment several times and picked out two extremely different results that represented the whole set of experiments. Out of 10 runs, 7 of the results had a congestion window size plotted over time similar to figure 7. In this figure we represent time as the relative packet number sent. Note that we are only plotting the first 5 seconds of packets for each experiment.

Figure 7 shows XCP constantly fluctuating its rate and under-utilizing the link. Zhang and Henderson claim in their analysis that while fine tuning the link capacity the MAC overhead needs to be considered. They know the capacity given their experiment is wired, however they wanted to estimate the proper framing and MAC overhead to properly set XCP's true capacity limit. They show that if they overestimate the overhead, which would be underestimating XCP's use able capacity, XCP will oscillate and under-utilize the link. This is similar to what we experienced in our wireless experiments, however it is far less predictable than in the wired medium. We believe this to be the case since it very hard to accurately estimate wireless link capacity since it is constantly changing. This would make it extremely hard to properly set the link capacity parameter.

For three of the ten experiments we saw a completely different behavior, where the capacity we had set was over-estimating the capacity of the wireless channel at the time

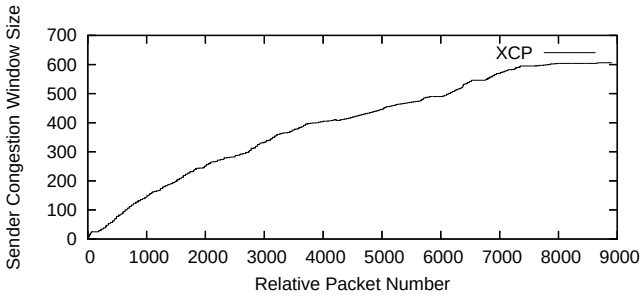

Fig. 8. This behavior was observed in 3 of our 10 stability analysis experiments, showing XCP overestimating capacity and filling up the queue, reducing performance.

of the experiment. This is shown in figure 8. When XCP overestimates the capacity, it will fill up the queue, and generate inflated feedbacks. This was also noted by Zhang and Henderson in their analysis of sensitivity to link contention. They set their links to half duplex and had multiple flows competing at a hub. We believe that we observed this when the two nodes were competing for the wireless medium, reducing the link capacity while doing so, causing XCP's capacity parameter to overestimate the actual capacity and fill up the queue.

This stability analysis is crucial to XCP's performance in wireless medium. Note that from the two figures, figure 7 and 8, the bandwidth achieved was 2827KB/sec and 2503KB/sec respectively. This shows to have a performance impact, and is why we believe our results fluctuated greatly in our analysis. Being that wireless is a shared medium with contention based on the number of nodes, we conclude that with the current state of the XCP implementation, estimating the capacity parameter properly for a number of nodes unknown would be impossible. Furthermore, if there were a fixed number of nodes competing for the medium, due to the possibility of collisions and variability in overhead of the MAC protocol from random exponential back-offs we still believe estimating the capacity parameter accurately would not be possible.

We conclude that XCP is very unstable when used in a wireless medium and would need improvements to its protocol to properly estimate capacity or modifications to make XCP less sensitive to the capacity parameter. This also has major implications for XCP's performance in a wireless medium. If capacity were estimated correctly and XCP could stabilize, we would expect XCP to outperform TCP in terms of throughput to a of much greater magnitude.

D. Fairness

One of XCP's major components is its fairness controller. Katabi et al have shown through their evaluation that XCP provides fairness greater than RED, CSFQ, REM, and AVQ. To evaluate XCP's fairness in a wireless environment, we have chosen to show the level of fairness when the capacity parameter is both underestimated and overestimated. By doing this we illustrate the sensitivity of the parameter to the fairness provided by the protocol.

For our first evaluation of fairness, we use two stations transferring directly between each other with thirty simultane-

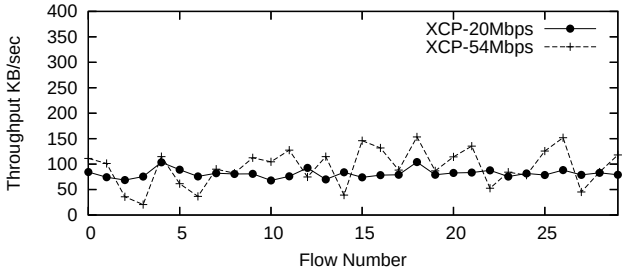

Fig. 9. XCP maintains a level of fairness when the capacity parameter is not overestimated for long term flows, 120 seconds each. When the parameter is overestimated, XCP's fairness controller breaks down.

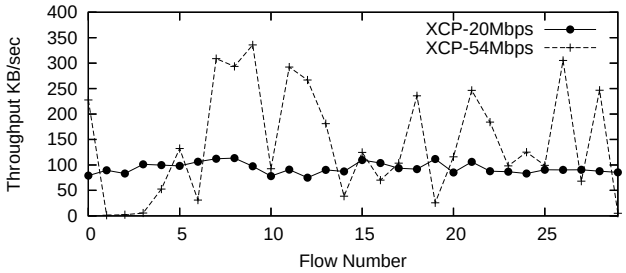

Fig. 10. XCP continues to maintain fairness when the capacity parameter is underestimated, but when the parameter is overestimated, our results show that for short term flows XCP's level of fairness is worse than long term flows.

ous flows, each being 120 seconds in length. This evaluation will show XCP's long term fairness. To show the effects of the capacity parameter, we run the experiment twice, underestimated it at 20Mbps and overestimated it at 54Mbps. The results we have obtained are shown in figure 9.

XCP is shown to maintain fairness in a wireless environment with long term flows when the capacity parameter is underestimated. When the capacity parameter is overestimated, XCP's fairness controller breaks down and gives greatly varying rates to different flows. This further illustrates XCP's sensitivity to the capacity parameter, sacrificing efficiency and fairness when overestimated.

To show XCP's short term fairness with the capacity parameter underestimated and overestimated we used 10 second flows to obtain the graph shown in figure 10. This graph shows that XCP's fairness breaks down to an even greater level with short term flows, than long term flows, when the capacity parameter is overestimated.

For further evaluation of XCP's fairness controller in a wireless environment, we introduced a third station to compete for capacity at the XCP enabled server. Our results show are shown in figures 12 and 11 where flows 1-15 belong to station one and flows 16-30 belong to station two. The data shows that contention for the shared wireless medium has a great effect on the fairness provided by XCP. In both figures, overestimating or underestimating the capacity parameter is shown to not have a great effect when there is contention in the medium. We speculate that the fairness inherent in the 802.11 protocol may reduce the effectiveness of XCP's fairness controller.

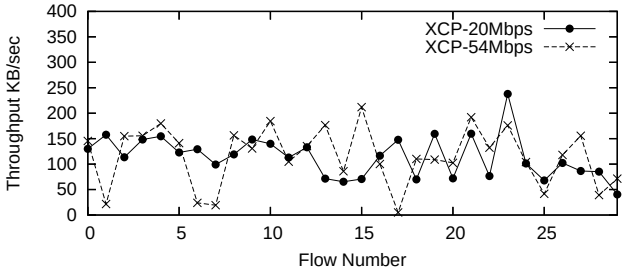

Fig. 11. XCP continues to maintain fairness when the capacity parameter is underestimated, but when the parameter is overestimated, our results show that for short term flows XCP's level of fairness is worse than long term flows.

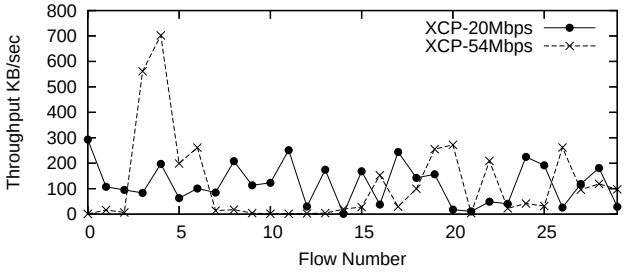

Fig. 12. XCP continues to maintain fairness when the capacity parameter is underestimated, but when the parameter is overestimated, our results show that for short term flows XCP's level of fairness is worse than long term flows.

VI. Speculations

We were unable to acquire enough machines to test out other topologies like start and mesh. Our current scenario is restricted in that there is only a single flow at given point of time through a node. It would be interesting to observe how XCP performs as compared to TCP while multiple flows share link capacity. We believe that XCP's fairness controller would allow a greater level of fairness over TCP, however, we speculate that this level of efficiency will be affected by the fairness built into the 802.11 MAC protocol. For further testing we would like to analyze XCP's level of fairness with the 802.11 MAC protocol in contrast to its fairness in a wired environment.

We also have our current implementation written to introduce loss at the kernel level, instead of at the hardware level. Since our packets are dropped at the kernel, the wireless card never sees the loss, and rate adaptation never occurs. If we could manage to introduce loss at the hardware level, the rate adaptation kicked in by the card might make a considerable difference. We hypothesize that the rate adaptation will affect XCP's stability to an even greater level than we have already shown in our stability analysis.

Our final speculation is on XCP's performance with asymmetric properties of wireless links. Balakrishnan et al [15] have shown TCP to have performance issues when dealing with asymmetric links such as cable modems and wireless networks. They have shown TCP to perform poorly where the reverse link is constrained, causing forward link underutilization from ACKs being dropped and TCP's ACK clocking mechanism, reducing the rate with large delays in ACKs from ACKs being queued.

VII. Conclusion

Our results have shown that XCP, although never designed for this purpose, can actually perform quite well with non-congestion related loss. We have also shown that due to XCP's sensitivity of the capacity parameter, and the inability to properly predict capacity in wireless networks, XCP will underutilize the link when both underestimating and overestimating the parameter.

Although we have shown that XCP can achieve 200% efficiency over TCP in certain cases of loss, XCP could actually perform to a much greater magnitude if the capacity parameter could be properly configured or if a modification to the protocol was made to be less sensitive to the parameter. For future work we would like to address this issue to develop a wireless version of XCP, increasing performance of XCP in lossy and non-lossy wireless networks.

We conclude that while XCP has properties which allow it to perform well in lossy 802.11 wireless networks, modifications need to be made to the protocol to reduce the amount of oscillations and increase its efficiency. Further work can also be done to analyze XCP's fairness with the 802.11 MAC protocol and XCP's performance with rate adaptation and asymmetric links.

Acknowledgment

We would like to thank Dave Andersen for not only providing us with the idea for the project, but also providing us support and hardware. We would also like to thank the Information Networking Institute for providing us with the machines and hall space necessary to complete the experiments.

References

[1] D. Katabi, M. Handley, C. Rohrs. *Congestion Control for High Bandwidth-Delay Product Networks*. ACM SIGCOMM 2002.

[2] Y. Zhang and T. Henderson. *An Implementation and Experimental Study of the eXplicit Control Protocol (XCP)*. IEEE Infocom 2005.

[3] P. Mosebekk. *A Linux implementation and analysis of the eXplicit Control Protocol (XCP)*.

[4] H. Balakrishnan, S. Seshan, E. Amir, and R. Katz. *Improving TCP/IP Performance Over Wireless Networks*. In Proceedings of ACM MobiCom, November 1995.

[5] G. Holland, N. Vaidya. *Analysis of TCP over mobile Ad Hoc Networks*. Proc. ACM MobiCom 1999, 1999, pp. 219-230.

[6] T. Dyer, R. Boppana. *A comparison of TCP performance over three routing protocols for mobile ad hoc networks*. Proc. of ACM MOBIHOC, Long Beach, CA, USA, 2001, pp. 5666.

[7] F. Klemm, Z. Ye, S. Krishnamurthy, S. Tripathi. *Improving TCP Performance in Ad Hoc Networks Using Signal Strength based Link Management*.

[8] H. Lim, K. Xu, and M. Gerla. *TCP performance over multipath routing in mobile ad hoc networks*. Proceedings of IEEE International Conference on Communications (ICC), vol. 2, May 2003, pp. 1064-1068.

[9] S. Floyd. *TCP and explicit congestion notification*. ACM Computer Communication Review, vol. 24, pp. 10-24, Oct. 1994.

[10] B. S. Bakshi, P. Krishna, N. H. Vaidya, and D. K. Pradhan. *Improving performance of TCP over wireless networks*. International Conference on Distributed Computing Systems, 1997.

[11] H. Balakrishnan, V. N. Padmanabhan, S. Seshan, and R. H. Katz. *A Comparison of Mechanisms for Improving TCP Performance over Wireless Links*. IEEE/ACM Transactions on Networking, Dec. 1997.

[12] K. Chandran, S. Raghunathan, S. Venkatesan, and R. Prakash. *A feedback based scheme for improving TCP performance in Ad-Hoc wireless networks*. Proc. of the International Conference on Distributed Computing Systems (ICDCS'98), Amsterdam, Netherlands, May 1998.

[13] L. Jinyang, C. Blake, D. De Couto, H. Lee, R. Morris. *Capacity of Ad hoc Wireless Networks*. Proc. ACM/ IEEE MOBICOM , Rome, Italy, July 2001.

[14] R. Jain, K. Ramakrishnan and D. Chiu. *Congestion Avoidance in Computer Networks with a Connectionless Network Layer*. SIGCOMM'88, Sept. 1988, 303-313.

[15] H. Balakrishnan, V. Padmanabhan, and R. Katz. *The Effects of Asymmetry on TCP Performance*. ACM Mobile Networks and Applications (MONET), 1998.