Interactions between TCP, UDP and Routing Protocols in Wireless Multi-hop Ad hoc Networks

Christian Rohner, Erik Nordström, Per Gunningberg, Christian Tschudin[b]

Uppsala University, Sweden and University of Basel, Switzerland[b]
{chrohner|erikn|perg}@it.uu.se, christian.tschudin@unibas.ch

Abstract

The achieved throughput in an ad hoc network is affected by many factors, including radio interference between hops, the ability of the routing protocol to react on topology changes and a complex interaction between the application and underlying protocols. This paper studies experimentally the impact of these factors on UDP and TCP throughput. Furthermore, when both TCP and UDP share some hops in an ad hoc network there is a complex interaction between the transport protocols as well as with the routing protocol. Our results show that this interaction results in significant UDP jitter, instable routes and significantly lower TCP throughput. We use a controlled real testbed for the experiments and a graphical tool that captures the interactions.

1 Introduction

It is well known that TCP suffers in a wireless environment because it incorrectly interprets packet loss as congestion. It is also well known that both UDP and TCP performance is reduced in a multi-hop 802.11 environment because of radio interference between hops. Simulation results by Holland et al. [5], show that the radio interference for two hops reduces the possible throughput to 50 percentage. For three hops the throughput is 33 percentage. This result was supported both by simulations and a theoretical model.

Another performance impairing factor is frequent routing updates. In ad hoc networks with mobile nodes, the problem for TCP is getting more severe since it is likely that packets get lost when a route is lost and the routing protocol needs to discover a new route.

Finally, it is also known that UDP is unfair to TCP when they share a common bottleneck link. TCP will back off and try to find a sending rate that adjusts to the rate of UDP. This adjusting time can be long depending on end-to-end delay and assumes some stability in available bandwidth. All these factors impact the performance of an ad hoc network and have been individually studied, but the interactions between them in a dynamic environment is not generally understood.

The intention with this paper is to demonstrate measurements results and to study unforeseen interactions between components at different layers that negatively impact the experienced application level performance in the ad hoc network. With our tools we can capture the series of transmissions involved and explain the timing of events relative to the node movements. We do this in a controlled experimental environment with the possibility to repeat experiments to understand the variance. The scenarios are carefully chosen to reflect the intended interaction between protocols, radio range, movement and prospective applications.

All scenarios are derivatives of the scenario shown in Figure 1. It comprises three nodes that have contact only with their adjacent neighbors. A low rate UDP flow, traversing two hops, is sent from node 2 to node 0. This scenario constitutes our baseline case, which remains static and is never changed in any of the scenarios. We then add a fourth node, sending an additional TCP flow to node 0. We also add dynamic routing and mobility. We do this to study the effect on the baseline case in terms of interference, contention for the wireless channel and buffer space. The low rate UDP flow is used to sample the wireless channel so that we can infer the interference from the fourth node. In theory, the expected result would be that the fourth node will have a limited

effect on the UDP flow, because TCP should adapt to the remaining bandwidth.

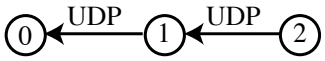

Figure 1. Base scenario: A static two-hop setting with a constant bit rate UDP flow.

From our measurement results we find that adding the fourth node interferes with UDP, inducing jitter and packet loss. Adding mobility and dynamic routing causes an average packet loss of up to 21.6% on the UDP flow. The results indicate that TCP and UDP contend for both buffers at nodes and radio transmission time. Both protocols suffer from significant loss as expected, and TCP backs off. Somewhat unexpected is that UDP losses are higher than could be expected on shared wireless links. This raises the question if TCP can adapt quickly enough to the environment where there is node movement and connectivity changes.

Another observation is that the routing protocol is also affected by the TCP flow, preventing the establishment of correct routes during extended periods. As TCP will back off and then start to probe again for the sustainable bandwidth, this potentially leads to long term oscillations overlaying the short term interactions among the protocols.

The paper is structured as follow. In section two we discuss related work. In section 3 we describe our experimental set-up, scenarios, experiments and results. We conclude with a section on discussion and further work.

2 Related Work

The TCP feedback loop is responsible for adapting the sender data rate in response to, e.g., congestion. However, this end-to-end congestion control mechanism has reduced efficiency in wireless networks because transmission is inherently broadcast. Furthermore, there are different and transmission rate dependent ranges for unicast transmission, broadcasts and interference [2].

Holland et al., examine in simulation the TCP performance over mobile ad hoc networks. They note that TCP suffers significantly in mobile, multi-hop scenarios, simply because TCP can not distinguish between link failures and congestion.

In [3], Gerla et al. study through simulation, the interactions between TCP and different MAC layers. Our work complements that work by also looking at other layers and different transport protocols.

Gupta et al. report in [4] on decreased TCP performance in the presence of interacting UDP flows. Their study is done in the ns-2 simulator on an artificial grid topology and with relatively high UDP data rate (800 Kb/s). Since UDP lacks rate adaptation and congestion control, a high data rate CBR flow will naturally congest the channel. Our work complement that work by performing real world measurements with UDP and TCP data flows in scenarios with increasing complexity. Moderate data rate UDP is used (typically streaming music) which – as we believe – is more realistic

Internet routers improve fair access to its resources by using Random Early Detection (RED). Xu, Gerla, Qi and Shu study in [8], TCP fairness in wireless ad hoc networks. They look at queuing policies to improve TCP fairness and show that the RED scheme fails, because flows compete not only for queue space but also for the wireless channel. They propose Neighborhood RED (NRED) where all individual queues of nodes within a neighborhood is treated as a shared distributed queue. Based on the notion of this distributed queue, a forwarding node can drop packets to increase TCP fairness.

3 Experiments

This section reports on real world experiments that examine the contention between TCP and UDP data flows. After providing some setup details, we discuss a sequence of increasingly more complex scenarios and the results of the following experiments.

All experiments were conducted indoors in a systematic way using the APE testbed [7]. APE provides a test environment that allows repeatability of experiments and provides support for extensive logging and analysis of experiment data. The APE testbed orchestrates the scenarios and provides verbose logging on the network interface and network layer. Standard laptops (IBM X31) with Lucent/ORiNOCO IEEE 802.11b network interfaces are used, transmitting at a fixed rate setting of 11 Mbit/s, without RTS/CTS. The interface used a standard MTU setting of 1500 bytes. Iperf and Ping were used to generate data. For UDP, 1470 bytes data

was sent[1], resulting in 1512 byte frames in the ether. This size was used to compare against the normal TCP segment size. Logged and time stamped data is time synchronized between nodes using periodic broadcasts of time information at a rate of one packet every 10 seconds. This time information is used to synchronize the experiment data collected from the different nodes.

The AODV-UU [1] implementation was used where dynamic routing was needed. AODV-UU was chosen because it is mature and has been interoperability tested.

3.1 Bandwidth Measurements

As a calibration and verification to prior work, we start by measuring the maximum achievable throughput for TCP and UDP data flows. We do this to establish the expected bandwidth for our baseline scenario, in single and multi-hop settings without mobility. For this experiment, four nodes are placed in a linear chain such that a node can only communicate with its adjacent neighbors. The UDP and TCP throughput over one to three hops are measured. The results are shown in Table 1. We differentiate in the one hop measurements between the two nodes located next to each other (short hop) and located in the periphery of each others transmission range (1 hop).

[Mbit/s]	short hop (10 cm)	1 hop (ca. 20 m)	2 hop	3 hop
UDP	5.64	4.85	3.60	2.01
TCP	3.71	1.68	0.78	0.62

Table 1. The maximum throughput is decreasing with increasing number of hops.

The UDP throughput is as expected higher than the TCP throughput, since UDP is one way traffic without acknowledgments that compete for transmission time over the shared channel. The maximum achievable TCP throughput over one hop is roughly a third of that of UDP. Acknowledgments going in the reverse direction is not as large as the data packets in the forward direction, giving more channel time for the data.

When increasing the number of traversed hops, the throughput decreases for each hop, in case of TCP following roughly an $1/n$ slope where n is the number of hops. This result matches the simulation results from Holland, et al. [5].

A noteworthy observation is the significant difference in one hop throughput between nodes that are placed next to each other (short hop, 10cm), and nodes that are located in the periphery of each others transmission range. In our indoor environment, that was a separation of 20 m. An analysis of the TCP experiments shows a slightly higher variation in the round trip time for separated nodes, indicating retransmissions on the link-layer as one reason for that artifact. TCP is more affected than UDP because of its own adaptation mechanisms that are not designed for fluctuating wireless environments.

3.2 Baseline Scenario: Static Multi-hop UDP

This experiment presents our baseline scenario shown in Figure 1. It consists of three nodes 0, 1, and 2 placed in line. Each node has connectivity only with its adjacent neighbors. End node 2 sends a constant bit rate UDP flow to node 0, over the intermediate node 1. As well as establishing the baseline performance, the experiment intends to examine the interference between the consecutive hops in the multi-hop path.

For each test run, the *offered* data rate (rate of data from the application) is increased, permitting to study the effect of interfering hops on the multi-hop path. Since the bandwidth at node 1's network interface is shared between the two "links" we expect the overall UDP throughput to decrease when the offered data rate at node 2 exceeds one half of the bandwidth at node 1.

Figure 2 shows the *received* data rate as a function of the *transmitted* data rate (rate of the data actually sent on the channel). It can be observed that the transmitted data rate for this two hop path achieves the best received data rate at around 3.6 Mbit/s. Increasing the offered data rate beyond that has a negative impact on the overall throughput, because of interference between the hops. Note that although we increased the offered data rate up to 10 Mbit/s, the *transmitted* rate never exceeded 5.2 Mbit/s.

After having determined the interference limited data rate over two hops, the UDP data rate was fixed at 192 kbit/s. This is well within the limit of link interference and also resembles the rate of an MP3 data flow. We expect such a low rate flow to have close to 100% delivery ratio in our baseline scenario with no interference.

The average UDP delivery ratio over five test runs was 99.2 %. It could be verified that packet losses oc-

[1] 1470 bytes is the default Iperf setting.

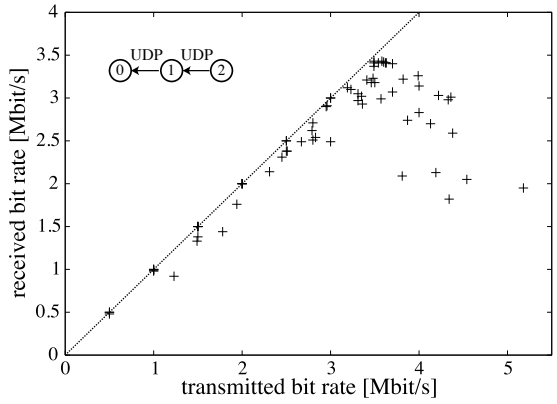

Figure 2. Received bit rate as a function of the transmitted bit rate in the two-hop scenario for offered bit rates between 0.5 Mbit/s and 10 Mbit/s (with 0.5 Mbit/s intervals in between). The line shows the optimal throughput when there is no packet loss.

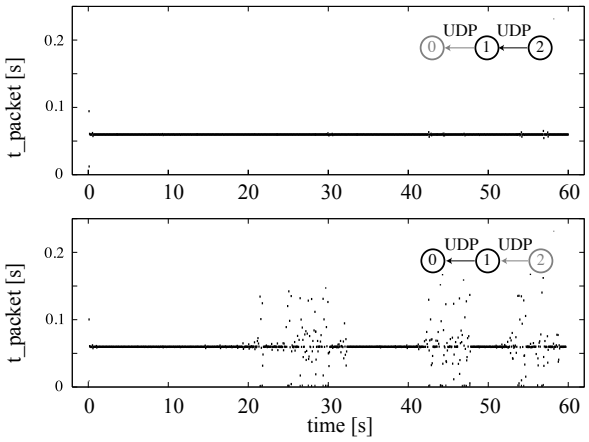

Figure 3. Example of UDP packet interspacing time in the static multi-hop scenario (192 kbit/s).

curred at the second hop link between node 1 and 0.

The interaction between the two hops is analyzed by looking at the UDP packet interspacing time. A variation in the packet interspacing time indicates either contention for medium access, link layer retransmissions, or packet loss. Figure 3 shows the packet interspacing time for the UDP flow between node 2 and 0 for a representative run. We observe that the second hop is more affected by interactions than the first link. However, as expected, the packet loss is low and packet interspacing small. In the following experiments we use these baseline results to study the effects on the UDP flow by adding an extra node, a TCP flow, dynamic routing and mobility.

3.3 Multi-hop UDP With an Interfering TCP Flow

In this scenario we extend the baseline scenario with an interfering TCP flow. An extra node 3 starts at the far left position in Figure 4, outside the transmission range of node 1. Ten seconds into the scenario it starts a TCP connection to node 0. After another ten seconds, node 3 starts moving toward node 0 where it stops its movements at time 38 but continues to send data. At this position, node 3 is within the transmission range of node 1 and hence will interfere.

Routing in this scenario is static. The purpose is to see how TCP adapts its send rate (if at all) when moving to a position where it is potentially more affected by channel contention from both node 1 and node 2, without an actual change in the path. We also want to see how the UDP flow from node 2 to node 0 is affected by the increased contention. We do this to better understand the effect of mobility and spatial placement of nodes.

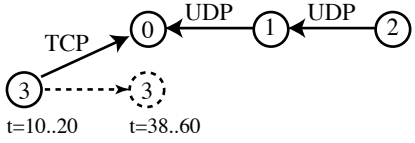

Figure 4. Scenario for two-hop UDP with an interfering one-hop TCP flow.

Over five test runs, the average UDP delivery ratio was 97.4 %. Although a slight decrease from the previous scenario, it does indicate that the UDP flow is not significantly affected by the interference from the TCP flow.

In Figure 5, we see the UDP packet interspacing time on the link between node 2 and 1, and 1 and 0 respectively, along with the TCP time sequence graph for the TCP flow between node 3 and 0, for one of the experiments. Although variations are apparent throughout all experiments, Figure 5 is representative for a typical test run.

As expected, the variation of the packet interspacing

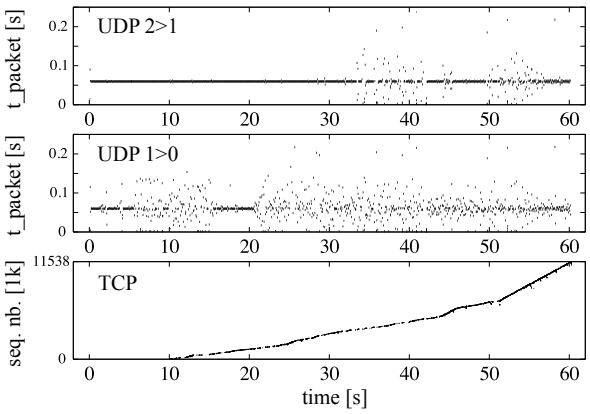

Figure 5. Example of UDP packet interspacing time and TCP sequence graph.

time between node 2 and 1 is more apparent after node 3 is in direct contact with node 1. It interesting though, that the packet interspacing time between node 1 and 0 tends to stabilize during the static periods (time 5..10, and after time 50 where basically the variation pattern from the first hop gets propagated).

The TCP flow itself is also affected. In the time sequence graph we observe some interruptions in the TCP flow of up to 500 ms, in particular during the movement of node 3 (time 20..38). Furthermore, TCP throughput is unstable even during the stationary phases at the beginning and end of the experiment. While the maximum achieved TCP throughput (1.63 respectively 3.37 Mbit/s) matches the one-hop figures of Table 1 where no UDP background traffic was present, the average throughput during these phases is significantly lower (0.89 respectively 1.66 Mbit/s).

An interesting observation can be made at time 50, when there is an immediate increase in the TCP throughput. This behavior is persistent in most of the experiments. It is not clear why TCP increases its throughput in this situation, at the same time as the fluctuations on the second UDP hop seem to stabilize. One possible explanation could be the capture effect [9]. Understanding the exact reasons for this particular behavior requires further investigation.

3.4 Multi-hop UDP Sharing Hops with a TCP Flow

This scenario, called “Roaming node”, extends the complexity of the baseline scenario by adding more mobility, multi-hop TCP, and dynamic routing using AODV-UU. The UDP flow, again, is sent from node 2 to node 0. Node 3 now starts out alongside node 0 at position A, as illustrated in Figure 6. Node 2 starts its UDP flow to node 1 simultaneously as node 3 starts a TCP file transfer to node 0 and starts moving towards position D. After 62 seconds it will reach position D and then turn back and move towards node 0 again. During this time, the TCP flow to node 0 is sent over a path that increases from one hop to two and three hops, and reduced back to one hop as node 3 is on the way back. Note that in this scenario, TCP and UDP have competing data flows going over the same intermediate nodes.

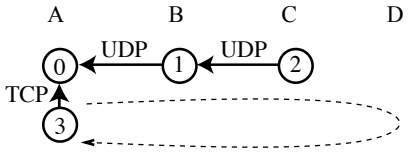

Figure 6. The Roaming Node scenario.

From separate analysis we know that there is one hop connectivity between node 3 and node 0 up to about time 25, followed by two hop connectivity up to time 50, and then three hop connectivity until time 92 when the route switches back to a one hop configuration until the end of the experiment. This is due to how AODV works; on the way back it will not optimize the route until a HELLO message is received by node 3 from the node 0.

The average UDP delivery ratio in this scenario decreased to 78.4% and the average TCP throughput to 0.93 Mbit/s [2]. A summary of the different scenarios in UDP deliver ratio and TCP throughput is shown in Table 2.

Figure 7 shows the UDP packet interspacing time and TCP sequence number graph for one of the experiments. It can be observed that the TCP data flow stalls in particular on the change from a one hop to a two hop configuration (time 25..30), and on the change from a two hop to a three hop configuration (time 40..50). There are also sporadic stalls during the three hop configuration (time 50..92). The different configurations are visible in the (slightly) decreasing slope of the time sequence graph. The switching of routes on the way back is much

[2] In the Roaming node scenario we experienced an outlier in one of our five test runs. Those results were therefore discarded and the average calculated over the remaining four runs.

Scenario	UDP Delivery Ratio	TCP Throughput
Static multi-hop UDP	99.2 %	-
Multi-hop UDP with TCP flow	97.4 %	1.34 Mbit/s
Roaming node	78.4 %	0.93 Mbit/s

Table 2. Average UDP delivery ratio and TCP throughput for the different scenarios.

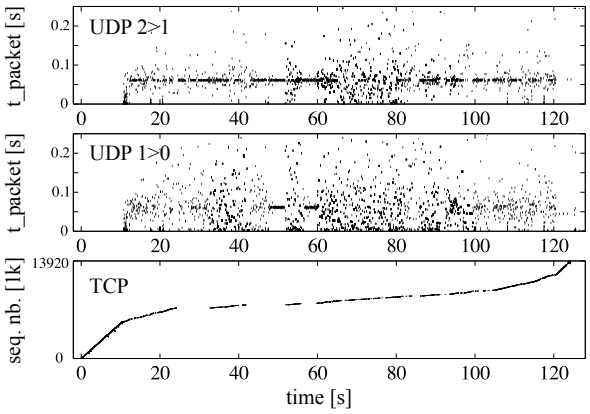

Figure 7. Example of UDP packet interspacing time and TCP sequence graph in the Roaming node scenario.

smoother. This can be explained by AODV's HELLO messages working more proactively on the way back, discovering a more optimal (shorter) route before the old one is gone.

We further observe increased UDP packet interspacing on the link between node 2 and 1 during three hop TCP connectivity (time 50..80). Increased UDP packet interspacing on the link between node 1 and node 0 is observed whenever we have progressing TCP traffic, but the UDP flow immediately stabilizes during TCP stalls. The increased packet interspacing is caused by the extra queuing delay on the intermediate nodes when the TCP flow also competes for buffer space.

An interesting observation can be made in the beginning of the experiment. It seems that when the TCP and UDP flows start simultaneously and in combination with dynamic AODV routing, the multi-hop UDP path between node 2 and 0 is broken. We see two different explanations for this (or likely a combination). Either TCP captures the channel, causing significant loss on the first hop UDP link (consequently no traffic is seen on the second hop link either). A more probable explanation, though, is that TCP's aggressive start impacts AODV's HELLO neighbor sensing and broadcast route discovery. The probing packets used in these mechanisms are broadcasted on the link layer, hence without acknowledgments or retransmissions. If these packets are lost, no path will be established. Not until time 10, after some movement will TCP back off and allow the UDP flow between node 2 and node 0 to resume. The reason for this is probably that node 2 was previously a hidden terminal for node 3, causing massive collisions at node 1. Only after node 3 moves within contention range of node 2, will it back off. The exact cause of this problem will be further examined in the next section, where we present a deeper analysis of protocol and link interactions.

3.5 Analyzing Protocol and Link Interactions with "Activity Plots"

The experimental analysis in the previous section raises many questions about the causes of the seemingly random and complex performance anomalies. It is necessary to study the interactions of protocols and link conditions at a much finer granularity in time to understand what is really happening. For this purpose we have developed an analysis tool to capture interactions between the protocols at different layers and for different nodes in a graphical way. This tool enables us to create "activity plots" from the collected data. Activity plots are a way to visualize and understand the timing of events. They complement the traditional flow based analysis with a more detailed analysis of activity on individual links. The idea is to present an experiment's complete set of events at a time granularity that is between single trace entries and aggregated performance figures.

An activity plot shows link activities on each nodes' wireless interface for all (or selected) links of a network in a single plot. A visual approach is better suited to comprehend the large amount of data collected during an experiment, providing for example an overview of the connectivity in the network and an easy way to spot connectivity problems. Coupled with traditional mea-

surements, such as a source node's TCP sequence number graph, it can be easier to identify and understand the spatial and temporal events that impact the performance.

We construct activity plots in the following way. During the experiment, all nodes record the successful reception of 802.11 frames, higher level events like route changes, and application layer specific data. The background time stamping application permits to synchronize the local traces after the test run and to merge the logs into a single experiment trace. From this trace file, we re-extract node-specific reception events and classify them according to their origin, creating a "link" view, and distinguish different message type in a graphical way. A link from node x to node y is denoted as $x \rightarrow y$. The links are laid out on the y-axis of the figure, while the x-axis represents time. Different activity events on a link are plotted as:

+ (large plus) unicast packet destined for node y.

+ **(small plus)** unicast packet overheard by node y.

◇ **(diamond)** broadcast packet. AODV route requests are highlighted with a large diamond.

We will now review some of our previous findings and discuss them using activity plots.

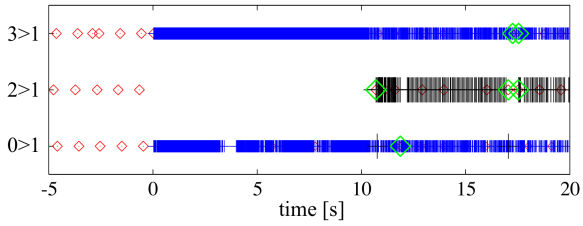

Figure 8. Activity plot showing the UDP flow starvation at the beginning of the experiment as seen by node 1.

An activity plot from the beginning of the Roaming node experiment from section 3.4 is shown in Figure 8. The plot shows all link activity as seen by node 1. In this scenario we know that node 3 is supposed to send TCP data to node 0 which is acknowledging the data packets starting at time 0. The activity plot shows that node 1 indeed overhears this traffic from nodes 3 and 0. At time 3, there seems to be a short interruption of the acknowledgments. A separate inspection of the link $3 \rightarrow 1$ however shows that these acknowledgments got to their destination (i.e., the TCP data flow is not interrupted).

The activity plot also shows that the UDP data flow sent from node 2 is not received until time 11. It further shows that the broadcast packets sent by node 2 (i.e., AODV HELLO messages), and received before time 0, stops being received up until time 11. From the activity plot we cannot conclude if node 2 was unsuccessful in contention for the wireless channel, or if node 1 did not receive the packets due to interference. If there would have been another node within connectivity of node 2 at that time, it might have overheard the packets. But from this information, we can not be sure. Clear though, is that the TCP flow affects the reception of HELLO messages. The route request visible at time 11 just before a burst of traffic on link $2 \rightarrow 1$, indicates that node 2 experienced problems to setup a route to node 1. The burst is caused by the transmission of buffered packets once a route is discovered. After the UDP flow is established, the packet interspacing of the overheard TCP data packets is increased. This is in line with the time-sequence diagram shown in Figure 7.

Another interesting part of the Roaming node experiment is the route change from one hop to two hops for the TCP flow around time 27. The activity plot in Figure 9 shows all activity originating from node 3, as perceived at node 0, 1, and 2 during the time when the route change occurs. A change in activity on the different links indicates connectivity problems on one of the links.

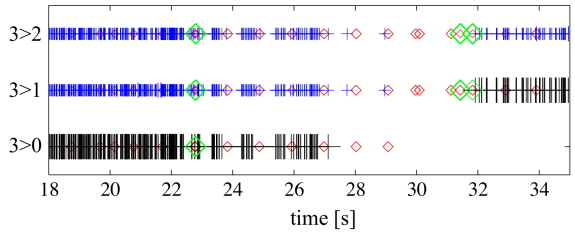

Figure 9. Activity plot showing all packets sent by node 3 illustrating the route change from one hop to two hops on the TCP link.

The route change going from a direct connection between node 3 and node 0, to a two hop route over the intermediate node 1, between time 27 and 32, can also be seen. The large plus symbols in Figure 9 indicate, before time 27, unicast packets on the link $3 \rightarrow 0$ and after time 32 on link $3 \rightarrow 1$.

Another interesting observation is that only broadcast packets are received by node 0 on the link $3 \rightarrow 0$ after time 27. After a while, also broadcast reception is impossible due to the increasing distance between nodes 3 and 0. The range where only broadcast but not unicast packets can be received is referred to as gray zones [6].

The main effect of the gray zone is on AODV, which during this time still assumes a valid route because it can receive broadcast HELLO messages, although no data traffic gets through. In the activity plot, it can be seen that nodes 1 and 2 actually overhear TCP retransmissions from node 3 while node 0 is in the gray zone of node 3. Only when broadcast reception stops will AODV time out its route. Not until after that time will TCP's retransmission trigger AODV to send a route request, searching for a new route (visible, for example, on the link $3 \rightarrow 1$ just before time 32). However, because of the exponential back off of TCP, this route discovery might not occur immediately after the gray zone. The TCP retransmission can be in a long timeout, effectively delaying the route re-discovery. This stall can be as long as twice the duration of traversing a gray zone, and severely worsens the effect of the problem.

4 Discussion and Conclusion

This paper has studied the effects on a low rate multihop UDP flow from a competing TCP flow. It was done through several experimental test scenarios where the TCP flow interfered, shared hops with the UDP flow and under mobility with dynamic routing. The purpose has been to study the interactions between the protocols at different layers.

The results indicate that although we use a moderate rate UDP flow, TCP's congestion control does not seem efficient enough to only have marginal impact on the other traffic in the network. When the two data flows do not share common links, we observe increased packet interspacing in the UDP flow, caused by jitter and to some extent packet loss. Instabilities in the form of short stalls are observed in TCP. Further analysis might show if TCP retransmissions are caused by lost packets or the high fluctuations in round trip time.

In the case where UDP and TCP share a common link, contention is significantly higher resulting in increased UDP packet loss and more significant TCP interruptions.

Dynamic routing, in particular when using broadcast neighbor sensing, adds another dimension of instability to ad hoc networking. Hidden terminals and channel capture effects cause otherwise stable routes to become unstable, simply because routing control messages are lost due to the competing data traffic. In our most complex scenario, the initial UDP flow experienced an average of 21.6% packet loss.

Our experiments show the complexity of the interactions in wireless multi-hop ad hoc networks. Therefore, we have developed an analysis tool to graphically analyze interactions between the protocols at different layers and for different nodes.

Acknowledgment

The authors would like to thank Laura M. Feeney for invaluable feedback on a prior version of this paper.

References

[1] The Uppsala University Ad Hoc Implementation Portal. http://core.it.uu.se/adhoc.

[2] G. Anastasi, E. Borgia, M. Conti, and E. Gregori. Wi-Fi in Ad Hoc Mode: A Measurement Study. In *Proceedings PerCom, Orlando (Florida)*, 2004.

[3] M. Gerla, K. Tang, and R. Bagrodia. TCP performance in wireless multi-hop networks. In *Proceedings of IEEE WMCSA'99 (to appear), (New Orleans, LA)*, February 1999.

[4] V. Gupta, S. V. Krishnamurthy, and M. Faloutsos. Improving the performance of TCP in the presence of interacting UDP flows in ad hoc networks. Unpublished.

[5] G. Holland and N. Vaidya. Analysis of TCP performance over mobile ad hoc networks. *Wireless Networks*, (8):275–288, 2002.

[6] H. Lundgren, E. Nordström, and C. Tschudin. Coping with communication gray zones in IEEE 802.11b based ad hoc networks. In *Proceedings of The Fifth ACM International Workshop On Wireless Mobile Multimedia (WoWMoM)*, September 2002.

[7] E. Nordström, P. Gunningberg, and H. Lundgren. A testbed and methodology for experimental evaluation of wireless mobile ad hoc networks. In *Proceedings of Tridentcom*, February 2005.

[8] K. Xu, M. Gerla, L. Qi, and Y. Shu. Enhancing TCP fairness in ad hoc wireless networks using neighborhood red. In *Proceedings of MobiCom'03, San Diego, USA*, September 2003.

[9] S. Xu and T. Saadawi. Does the ieee 802.11 mac protocol work well in multihop wireless ad hoc networks? *IEEE Communications Magazine*, 39(6), June 2001.