Interactions Between TCP and the IEEE 802.11 MAC Protocol

Rui Jiang Vikram Gupta Chinya V. Ravishankar

Computer Science and Engineering Department
University of California, Riverside

Abstract

The IEEE 802.11x MAC protocol, the de facto standard for wireless LANs, includes a distributed coordination function (DCF) mode usable for ad hoc network architectures. The Transmission Control Protocol (TCP) is commonly used on top of this MAC for reliable transport. TCP's congestion control scheme assumes highly reliable frame transmission at the link layer. Unfortunately, this assumption does not hold in multi-hop wireless network scenarios, leading to severe performance degradation. In this paper we study the interactions between the TCP and 802.11b MAC protocols, and examine the effects of different TCP and 802.11b parameters on the throughput achieved. Using simulations and analysis, we highlight the causes of performance degradation when TCP is used over 802.11b MAC in a multi-hop wireless scenario.

1. Introduction

A wireless ad hoc network is an autonomous system of routers (and associated hosts) connected by wireless links. Each node in such a network can also act as a router, forwarding packets on behalf of other nodes. The wireless links share the communication medium, so an efficient access control mechanism must coordinate the use of this shared resource. Medium access control design for wireless LAN has been an active area of research recently, and a number of MAC protocols [13, 5, 15] have been proposed. However, only a few of these explicitly account for the multi-hop requirements of ad hoc networks.

The IEEE 802.11 MAC protocol [12] is the current de facto standard for wireless links. It includes a "Distributed Foundation Wireless Media Access Control (DFWMAC)" to support ad-hoc and infrastructure LANs. Many applications in ad-hoc networks depend on a reliable transport protocol. TCP is now the prevalent transport protocol in IP-based networks, and future ad hoc networks are likely to adopt TCP. It is important to understand the behavior of TCP when coupled with IEEE 802.11 MAC protocol in an ad hoc network.

It has been known that TCP does not work well in wireless networks [10]. Wireless channels are inherently noisy, so that packet losses are more frequent in wireless networks than in wired networks. TCP does not differentiate between congestion-related packet drops and transmission failures at link layer. TCP treats all packet losses as indicators of network congestion, and triggers the congestion control mechanism. Consequently, transmission failures at the MAC layer causes the activation congestion control in TCP, and reduces the throughput. Several mechanisms [4, 1, 6] have been proposed to address this problem. However, these ideas are focused on cellular network architectures, in which a base station is available and the wireless link is used only for the last hop.

The problem is more complex in multi-hop wireless networks, as there are no base stations, and any node may act as a router. Previous work [9] has shown that TCP performance degrades in the multi-hop scenario under various MAC protocols like CSMA, MACA and MACAW. Furthermore, when TCP is coupled with IEEE 802.11 there are problems like throughput oscillation as indicated in [20].

Most previous work attributes TCP performance degradation to link layer unfairness and poorly designed backoff policies. Some new MAC protocols that address these issues have been proposed [11, 17]. However, the behavior of TCP itself or the interaction of TCP with the MAC protocol have not been sufficiently explored. As our work shows, TCP parameter settings also influence the throughput achieved by applications. TCP parameters are fine-tuned for wire-line networks, but that ad hoc networks differ significantly from wire-line ones. The mechanisms and policy employed by TCP may turn out to be unsuitable in ad hoc networks.

In this work we investigate the interaction between TCP and IEEE 802.11 MAC protocols through simulations. We first show that TCP does not work well with 802.11 MAC in multi-hop wireless networks. We then examine the effect of different parameters of TCP and 802.11 MAC protocols

Proceedings of the DARPA Information Survivability Conference and Exposition (DISCEX'03)
0-7695-1897-4/03 $17.00 © 2003 IEEE

on the performance of TCP throughput under different network topologies. By analyzing the trace files obtained from the simulations, we identify the causes of TCP throughput degradation. Further, we provide insights into the MAC protocol behavior that accentuates the problems.

2. Overview of IEEE 802.11 MAC

The IEEE 802.11 MAC protocol [12] defines two different access methods; a distributed coordination function (DCF) and polling based point coordination function (PCF). In ad hoc networks, the DCF feature is used.

The DCF access is basically a carrier sense multiple access with collision avoidance (CSMA/CA) mechanism. When a node wants to transmit a frame, it senses the medium. If the medium is busy, it defers this transmission. If the medium is free for a specified time, called the distributed inter-frame space (DIFS), the node is permitted to transmit. In order to avoid collision due to the hidden terminal problem [5] the node first transmits a Request To Send (RTS) control frame. The destination node responds with a Clear To Send (CTS) control frame. Both RTS and CTS frames include the duration of the transmission that will follow the RTS-CTS exchange. All nearby nodes receiving either the RTS or the CTS frame defer their pending transmissions for this duration. This deferral is referred to as virtual carrier sensing as it "senses" the medium through exchange of frames at the MAC layer. Once a successful RTS-CTS frame exchange takes place, the data frame is transmitted. The receiving node checks the received data frame, and upon correct receipt, sends an acknowledgement frame. If the sending node fails to receive the acknowledgement frame, it assumes that the data frame was lost. It backs-off and attempts a retransmission. After repeated failures to get the frame across, it simply drops the frame. Thus, although the introduction of RTS-CTS-DATA-ACK frame format makes the transmission more reliable, there is still the possibility of transmission failure. Such failures are more frequent in ad hoc networks than in wire-line or single hop wireless LANs.

3. TCP over IEEE 802.11 MAC protocol

The IEEE 802.11 MAC protocol is designed to provide efficient and reliable frame exchange over the shared wireless channel. Consequently, this protocol has a rather limited view of the network conditions. The information available to it is the status of its direct neighbors. In contrast, TCP observes the network conditions at a higher level, and its view of the network conditions is independent of the underlying protocols. Unfortunately, TCP assumes that packet loss between the end to end hosts is always due to congestion. That is, it assumes a reliable link layer, a valid assumption in wire-line networks, but not over the 802.11b MAC. Consequently, TCP's reactions to specific network conditions is sometimes incorrect.

In the following sections, we study the problems when TCP is coupled with the IEEE 802.11 MAC protocol. First we introduce the simulation setup.

4. Simulation setup

We use the NS2 network simulator from LBNL [14] with wireless extensions from CMU [8]. The extensions include a set of mobile ad hoc network routing protocols and 802.11 MAC protocol.

The link layer of the simulator implements the IEEE 802.11 standard MAC protocol under DCF. All nodes communicate over identical half-duplex wireless links with a bandwidth of 1Mb/s. For our simulations the effective transmission range is set to 250 meters. We assume an interference range of 550 meters, so that nodes within this distance of a transmitting node will sense the medium as busy. Each node has a 50-packet link layer buffer queue managed in a drop-tail fashion. The scheduling of packet transmissions is FIFO. The routing protocol used in the simulation is AODV [16].

We consider two types of network topologies. First, we examine string topologies with different number of hops, to serve as a simple yet effective model for multi-hop connectivity scenarios. For a more realistic scenario we study two strings that share a node, and examine media access at the shared node, and its effects on the throughput of both TCP connections.

The distance between any two neighboring nodes is set to 200 meters, so that a node may communicate only with it's closest neighbors. The interference range (under normal channel conditions) of a node is slightly greater than twice its transmission range (550m in this case). By spacing all nodes equally, we are guaranteed that the interference conditions are uniform. Nodes are not mobile during the simulation, so we do not address the link failure problem caused by node mobility.

5. Interaction between TCP and IEEE 802.11

TCP and 802.11 protocols allow some parameters to be adjusted for optimal operation. The current TCP protocol implementations generally assume that round trip times can be measured accurately, that links are stable, and that their capacities are fixed. However, in a multi-hop wireless environment, such assumptions do not hold due to dynamic link formations. Thus, it becomes important to observe the performance under various values of TCP parameters and find an optimal setting for an ad hoc networking environment.

Proceedings of the DARPA Information Survivability Conference and Exposition (DISCEX'03)
0-7695-1897-4/03 $17.00 © 2003 IEEE

The IEEE 802.11b MAC protocol is primarily designed for wireless LANs with access points, although it supports ad hoc functionality. Thus, the parameter settings in MAC protocol may not be ideal for multi-hop environments. Further, when this MAC is coupled with TCP, the resulting interactions can be quite complex. Certain TCP mechanisms and policies that improve performance in wire-line networks may have adverse effects when running over the 802.11b MAC protocol. Similarly, the MAC level unfairness may be increased by short (in hops) TCP connections that are able to send more packets over the medium. Furthermore, TCP and 802.11b both provide some similar functions and have some similar mechanisms. For example, TCP guarantees end-to-end reliability, but the 802.11 MAC also employs link level acknowledgement to guarantee point-to-point reliability. Also, TCP has a back-off policy when it encounters packet loss, and the 802.11b protocol uses a similar method to avoid channel access conflict.

In this section, we study the interactions between TCP and the IEEE 802.11 MAC, and the effects of different parameters on TCP throughput. We revisit some existing problems and take further steps to look at the detailed reasons that cause the problems, and then provide some possible ways to alleviate the problems.

5.1. TCP congestion window size

In this section we investigate the effect of TCP congestion window size on TCP throughput. We simulate string topologies with different numbers of hops, as shown in Figure 1. A TCP sender and TCP receiver is chosen and a TCP session is established. There is no background traffic in the network.

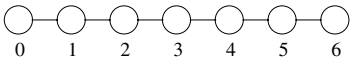
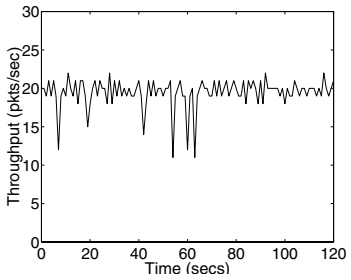
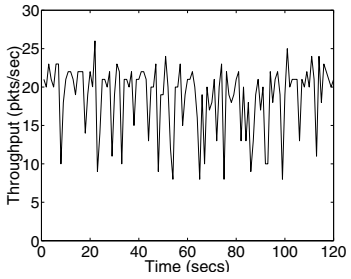


Figure 1. String topology.

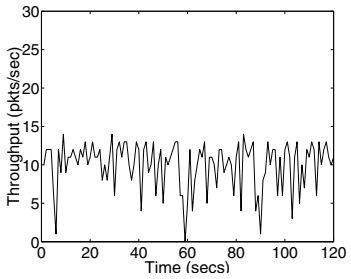
The TCP congestion control mechanism continuously increases its congestion window until it consumes all the available bandwidth or detects a packet loss. The Reno version of TCP also employs a fast recovery mechanism to achieve stable throughput. In this simulation we setup single TCP session. Since there is no background traffic contending with this TCP session, we might expect to see stabilized throughput. However, we do not observe a stable throughput. When the maximum congestion window at the TCP sender increases, TCP throughput fluctuates, as found in previous work [20]. Moreover, we find that the congestion window size that causes such oscillation depends on the number of hops in the TCP connection. The longer (in hops) the connection, the smaller the congestion window size that triggers the fluctuation effect.



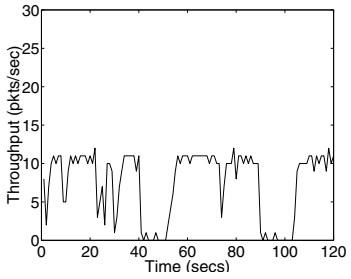
(a) 3-hop connection, W=4



(b) 3-hop connection, W=16



(c) 6-hop connection, W=8



(d) 12-hop connection, W=4

Figure 2. Oscillations for string topology.

Proceedings of the DARPA Information Survivability Conference and Exposition (DISCEX'03)
0-7695-1897-4/03 $17.00 © 2003 IEEE

Our simulation results are shown in Figure 2, which plots the throughput of TCP connections with different number of hops and various TCP congestion window sizes (W). As the current version of NS2 only supports fixed-size TCP packets, we fix the packet size at 1460 bytes and measure throughput in terms of the number of packets received. The throughput is measured over a one-second interval. The transmission time for a 1500 byte packet over 12 hops is approximately 0.15 secs. Our choice of 1 second aggregation interval is based on this value. Choosing a larger value may occlude the analysis. It should be noted that we do not intend to justify the need for smooth TCP throughput at any time granularity. Our objective is to simply observe the oscillatory behavior of TCP. However, measures taken to smoothen the TCP throughput over 1 second interval may suffice most application's needs. Each simulation runs for 120 seconds. Figure 2 illustrates the fluctuation effect for 3-hop, 6-hop, and 12-hop connections. These are representative of different deployment scenarios for multihop wireless networks. We conduct the experiments for different congestion window sizes and hop counts. Our results show consistently similar pattern, so we present only some of them here.

From Figure 2(a), we can see that for a 3-hop connection, the throughput is relatively stable within the 120s simulation period when the maximum size of TCP congestion window is 4. The TCP window size here is counted in number of packets. However, when this value is increased to 16, we see severe oscillations as indicated in Figure 2(b). Figure 2(c) plots the throughput of a 6-hop connection. It shows that with the value 8, we see the oscillation of throughput. The throughput even reduces to zero at certain times. In Figure 2(d), the situation worsens. For a 12-hop connection, the throughput reduces to zero for a 14 second duration even for a low value (4) of TCP congestion window size.

Given the above observations, it is clear that the oscillation effect can be alleviated by reducing the TCP congestion window size. Previous studies show that the optimal window size is 1/3 path length [19] or even 1 in most practical situations [18]. The reason that such a small congestion window generates the highest throughput is partially due to the MAC protocol's contention for the channel. For example, in a 3-hop connection, almost every node is in the interference range of all others, which means only one node can transmit at any time. (The details of link layer conflict are discussed later.) Limiting the TCP congestion window size is acceptable for a short (1-4 hop) connection since its round trip time is also short. Thus even a small window can keep the "pipe" full. However, for a relatively longer connection, a smaller window limits the number of packets in transit, which means fewer nodes will have packets available to transmit. A smaller window size can thus reduce resource utilization for longer connections. However, we first need an in-depth understanding of the causes of throughput oscillation.

We simulate only one connection in a static string topology, so there is no contention for the bandwidth with other flows and no mobility related link failures. Hence, there is no a priori reason to expect throughput oscillations, with long periods of zero throughput. Our trace files indicate that the problem originates at the link layer, and occurs when a node fails to reach its neighbor, even though they are in transmission range of each other. If a packet is not acknowledged, the 802.11b standard mandates the sending node to attempt 7 retransmissions for a short packet (RTS) times and 4 for a long (data) one. If a packet still cannot be sent across, a link failure is to be inferred. This inference in turn triggers a route failure at the network layer protocol (AODV). If the sending node is an intermediate node, it drops all the packets destined to go over that link, and reports a route failure to the source of the packet.

In a carrier sense wireless network, the interference range is typically larger than the communication range. The signal fading pattern of the WaveLAN wireless system that NS2 based on is such that the interference range is slightly more than two times larger than the effective communication range. Thus, in our simulation topologies, one node can sense the signal sent from two hops away but can only communicate to its directly adjacent neighboring nodes.

A typical collision condition is as follows. Consider the transmission of an RTS or a data packet from node 1 to node 2. Node 2 may receive the packet correctly but may be unable to send the corresponding CTS back to node 1. This problem can occur, for example, if node 4 is sending to node 5. During the transmission between nodes 4 and 5, node 2 can sense the signal from node 4. Even though node 4 has sent RTS to reserve the channel and node 3 responds with CTS indicating the length of the transmission to its neighboring nodes, node 1, being two hops away, cannot correctly receive this CTS. When node 4 is transmitting, node 1 will sense the channel as free and try to send its own packet. Node 1 fails to get response from node 2. Failing to receive a CTS from node 2 after the specified number of retransmissions, node 1 quits and assumes link breakage. At this point, a route failure is triggered, and the source node starts a route discovery process. Until a route is found, no data can be sent. If the route discovery process is not quick enough, the TCP sender may timeout, causing it to enter the slow start phase.

If an RTS-CTS mechanism is employed at the link layer, link utilization may be as low as one in four hops. Consider the 10-node string connection shown in Figure 3. The dotted circles denote the approximate interference range of each transmission. A transmission from node 1 interferes with node 3, which cannot simultaneously communicate with node 4. Similarly, a transmission by node 4 may cause

Proceedings of the DARPA Information Survivability Conference and Exposition (DISCEX'03)
0-7695-1897-4/03 $17.00 © 2003 IEEE

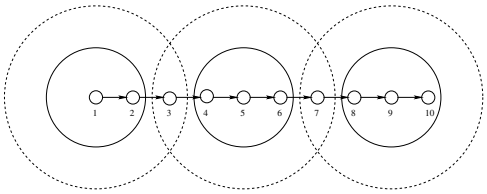


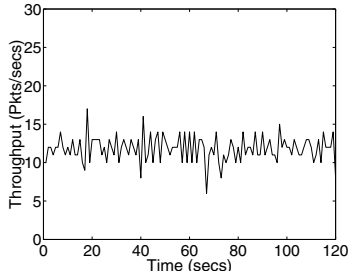
Figure 3. Communication and Interference ranges.

a collision at node 2. Thus, links 1-2, 5-6 and 9-10 represent maximum possible concurrent channel usage. For a string connection of n nodes, the maximum number of simultaneous transmissions possible is $\lceil (n-1)/4 \rceil$. If link 4-5 or 8-9 is active, only 2 simultaneous transmissions are possible. This analysis ignores the flow of TCP-ACK packets. An increase in number of nodes in the string may not change the number of simultaneous transmissions. However, it does permit a larger number of choices for non-interfering transmitters, increasing the likely channel utilization. Hence, using a TCP window size higher than $\lceil (n-1)/4 \rceil$ may not be useful.

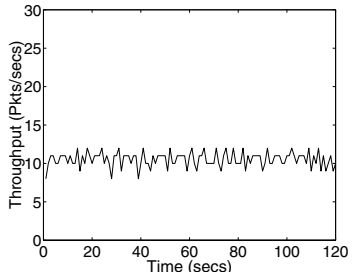
For analysis purposes, consider a 6-hop connection with maximum TCP congestion window set to 8 packets. During the 120s simulation period, 1179 packets were successfully received by the TCP receiver. However, the TCP sender sent out 1278 data packets, so that 98 packets, or 7.7 percent, are dropped during transmission. At the same time, 38 out of 1179 TCP ACK packets were dropped. In TCP Reno, with fast recovery mechanism, when a packet loss is detected, the congestion window is halved. Consecutive packet loss can reduce the congestion window sharply. The 7.7 percent packet loss rate causes the congestion window to change frequently, This explains the observed oscillation in throughput. Within 120 seconds simulation period, we also observe that route failure is triggered 21 times. We believe route failures to be the primary cause of the long pauses in TCP throughput observed.

Let us now compare the above connection with the maximum TCP congestion window reduced to 4 packets. During the simulation time, only 45 out of 1275 TCP data packets, or 3.5 percent, are dropped. The route failure event occurs only 5 times. These figures are significantly lower than that in the previous case. Thus, the resulting oscillation in throughput is significantly reduced.

Given the above results, it is clear the problem results from interaction between TCP and 802.11 protocol. When TCP increases its congestion window and sends out more packets, the probability of link layer collision(s) and transmission failures increases. Frequently, this leads to link failures at MAC sublayer, route failure at the network layer, and timeout at TCP sender. At the very least, TCP reduces



(a) 6-hop connection, W=8, Retry limit = 14/10



(b) 12-hop connection, W=4, Retry limit = 14/10

Figure 4. Increased retry limit.

the congestion window by half in reacting to every packet loss. When the congestion window is small, fewer packets are sent out to the medium, and the probability of failures at MAC sublayer is low. This explains why we see stable throughput if the TCP congestion window is limited to a small value, and oscillations otherwise.

5.2. Retry limit in 802.11 MAC

We have observed that if a response to a frame transmission is not received after the specified number of retries, the packet is dropped and link breakage reported. Also, the node drops all the packets destined to the same node in its IFQ. In our simulations, all nodes are static, so all packet drops are due to collision with neighboring nodes. Under the above scenario, an improvement for MAC protocol would be to let the sender retransmit an increased number of times. Therefore, if we increase the retry limit, we expect to reduce the chances of a packet being dropped at the link layer. Furthermore, we expect a reduction in the number of route failures and oscillatory behavior. In many cases such an improvement will also increase overall throughput.

In this section we look into the effects of increasing the MAC protocol's retry limit. We increase the retry limit for

Proceedings of the DARPA Information Survivability Conference and Exposition (DISCEX'03)
0-7695-1897-4/03 $17.00 © 2003 IEEE

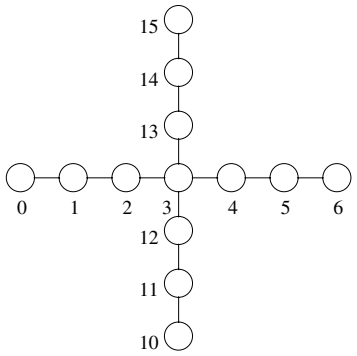


Figure 5. Cross connection topology.

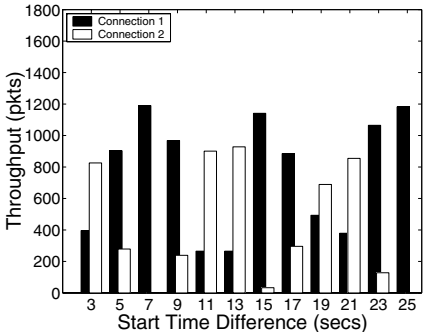


Figure 6. Capture effect with start time.

short frames from 7 to 14 and for long frames from 4 to 10. We repeat our earlier experiments for 6 and 12 hop connections. Figure 4 shows the experimental results for variation in throughput with these parameters. Compared with Figures 2(c) and 2(d), we find much lower throughput oscillation than with retry limits 7 and 4. The aggregated throughput is also increased. For example, for the 6-hop connection, total packets received increases from 1179 to 1396, which a 18 percent increment. For a 12-hop connection, the throughput improves by 39 percent, with total packets received increasing from 896 to 1243.

The current retry limits appear to be reasonable for wireless LANs with a base station, as every node is within the interference range of all other nodes. With a low retry limit, the node can detect link breakage early. If the retry limit is too high, it could end up wasting resources as a link failure due to mobility will take longer to be detected. Another drawback of a high retry limit is that the transmission of other packets in the link layer queue (IFQ) is delayed due to the FIFO scheduling of IFQ. Overall, this results in longer end-to-end delays. In many cases in multi-hop wireless networks, the cost of unnecessary packet drop may far outweigh the drawbacks of higher retry limits. This would be especially true in a low mobility situation. Thus, one of the possible ways to improve throughput of TCP based applications is by increasing the MAC protocol retry limit.

5.3. Capture effect at the MAC layer

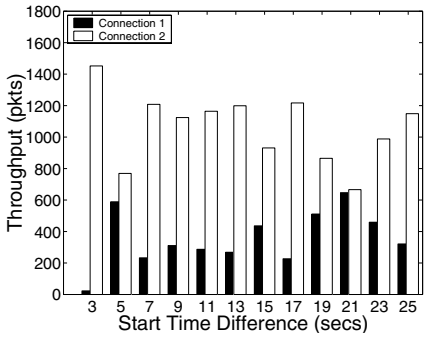
Previous studies [18, 19] show that IEEE 802.11 MAC suffers from a severe channel capture effect, causing an unfairness problem. Channel capture stems from the binary exponential back-off mechanism of the 802.11 MAC protocol. Binary exponential back-off favors the last successful node. The capture effect causes the most active connection to dominate the shared channel. Thus, when several TCP connections share the bandwidth, the connections starting early or more heavily loaded ones may have a higher probability of capturing the channel.

In our simulations we studied this effect for two connections that share a single intermediate node. The topology is shown in Figure 5. Connection 1 is from node 0 to node 6, and connection 2 is from node 10 to node 15. Our simulations suggest that under heavy load conditions starting early cannot guarantee that a connection will dominate the shared channel. Even if one connection has much higher data rate than the other, it may not dominate the channel under certain conditions. The length of the connection has more prominent effect on capturing the channel. A short connection (in hops) tends to dominate over longer connections and captures more bandwidth.

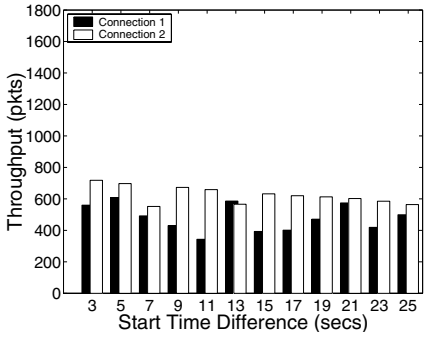
Figure 6 compares the overall throughput of the two crossed connections. In the simulation, the two TCP connections have the same load and the congestion window was not limited, which means that it can increase to the default maximum value specified by TCP protocol. The first connection starts early, and we simulate the scenarios where the second connection starts from 3 seconds to 25 seconds later than the first connection. The simulation time is set to 120 seconds. The duration of the second TCP connection is smaller than the first. If the channel is evenly shared between the two connections, we would expect to see connection 1 having a larger aggregate throughput than connection 2 in all scenarios. However, the results show that even if connection 2 starts later and runs a shorter time, it too has a chance to capture the shared channel.

In Figure 6, we notice that in some situations, connection 2 is completely dominated by connection 1, but in other situations, connection 2 gets more bandwidth than connection 1. In our experiments, all the parameters of the two connections are identical, except for their start times. So the difference in the throughput distribution is affected only by the start time of connection 2. The TCP receives data at a constant bit rate from the application. We simulate heavy load conditions, i.e., TCP has packets to send at all times. Thus, if there is no contention from other flows, every TCP connection will attempt to exhaust the available bandwidth at all times.

Proceedings of the DARPA Information Survivability Conference and Exposition (DISCEX'03)
0-7695-1897-4/03 $17.00 © 2003 IEEE



(a) 6-hop and 4-hop (50kbps)



(b) 6-hop (20kbps) and 4-hop (10kbps)

Figure 7. Capture effect with path length.

In the simulation topology, the only node shared by the two connections is node 3. Whether connection 2 is able to capture the channel or not is determined by the state of connection 1 when connection 2 starts. This state includes the current TCP congestion window size for connection 1 and the value of back off timers at the shared node and nearby nodes. If connection 1 is in a slow start phase, and has a small congestion window, connection 2 will have a high probability of capturing the shared channel at the intersecting node. Such a scenario can occur when there are only a small number of packets for connection 1 in transit, i.e. connection 1 does not use all available bandwidth at the shared node. If connection 1 has a large congestion window at the time when connection 2 starts transmission, more packets from connection 1 will be in transit. Thus, connection 2 has little chance of interrupting it and capturing the shared channel instead. Connection 1 will increase its congestion window until packet loss occurs. At this point, TCP decreases the congestion window sharply (by half). This will permit another connection to get the shared channel. Thus, TCP's congestion window oscillates continuously. The total throughput of connection 2 shows some randomness depending on the network conditions when it starts. In some extreme cases, one connection can completely dominate the other.

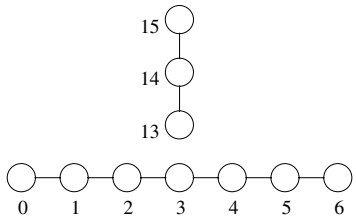


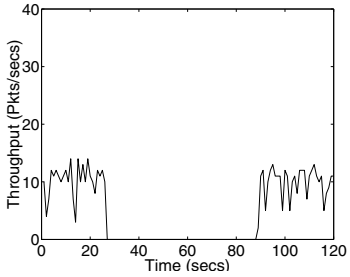
Figure 8. Neighborhood capture topology.

The length (in hops) of the connection plays a more important role in determining the TCP throughput than the connection start time and traffic load. Consider two TCP connections that have same traffic load (heavily loaded in this case) but connection 2 being shorter than connection 1. The shorter (in hops) connection (2) captures the shared channel in most of the cases independent of the starting times.

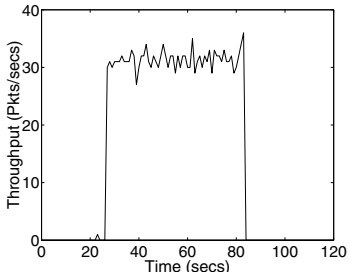
Our experimental results are shown in Figure 7. Figure 7(a) is for two connections carrying the same data rate of 50kb/s but of different lengths. The first connection (1) is from node 0 to node 6 while the second connection (2) is from node 11 to node 14. Connection 2 is only 4 hops long and thus, shorter than connection 1. Again, only node 3 is shared by the two connections. Comparing these results with those in to Figure 6, we observe that connection 2 dominates the shared channel in all cases. In a second experiment, we set different data rate for the two connections. Connection 1 has a data rate 20kbps, while connection 2 has a data rate of 10kbps, which is only half the rate of connection 1. Furthermore, connection 1 starts earlier than connection 2. However, the results in Figure 7(b) show that even when connection 2 has a lower rate and lasts for a shorter duration than connection 1, its overall throughput is still larger than that of connection 1.

The above results can be explained by noting that connection 2 has a smaller round trip time (RTT) value than connection 1. A short RTT value means successfully received data packets can be acknowledged faster, and the TCP congestion widow also grows quickly. So the packets from the short connection pass the shared node more frequently, and the short connection has a better chance at capturing the shared medium. The capture effect is more pronounced as the difference between the lengths of two connections increases. If there is a very short connection near by, a long connection can be completely dominated, even if the short connection does not have a node shared with the long connection. Such a scenario can occur when one of the nodes along the short connection is in the inter-

Proceedings of the DARPA Information Survivability Conference and Exposition (DISCEX'03)
0-7695-1897-4/03 $17.00 © 2003 IEEE



(a) 6-hop connection, node to node 6



(b) 2-hop connection, node 13 to node 15

Figure 9. Neighborhood capture effect.

ference range of the long connection.

Figure 8 shows such an example. Connection 1 is 6 hops long, from node 0 to node 6, while connection 2 is from node 13 to node 15, which is only 2 hops in length. Although connection 1 starts earlier, as soon as connection 2 starts, its throughput reduces to zero until connection 2 stops. The results of the experiment are shown in Figure 9. In Figure 9(a), connection 1 starts first and maintains a stable throughput when it is the only data flow in the network. As soon as connection 2 starts at 20th second and captures the channel after that, as shown in Figure 9(b), connection 1 stops completely, It resumes when connection 2 finishes its transmission.

5.4. TCP packet size

During a transmission, nodes within the interference range are "captured", i.e, they cannot respond to any communication requests or transmit. The duration of such a capture period is determined by the length of the data packet being transmitted. Nodes in the network that attempt to communicate with the captured nodes will infer the link to be broken after certain number of attempts. The duration between successive retransmissions is governed by the exponential back-off algorithm in IEEE 802.11 protocol. Thus, larger packet size can lead to increased link failures at MAC layer. Similarly, a small packet size can reduce duration of capture, resulting in frequent opportunities for channel access. However, a small packet size also increases the control overhead and can increase collisions at the link layer. In this section we examine the trade-off between increasing the packet size and the corresponding increase in false link failures.

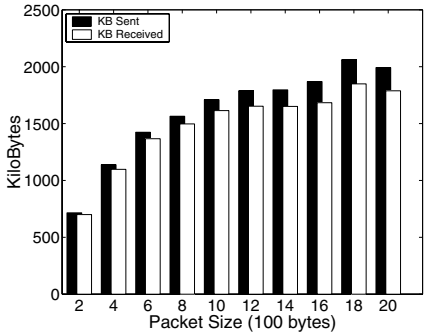
We observe the variation in TCP throughout with packet size. The simulations are conducted for a single TCP connection on a string topology. For multiple connections, aggregate throughput can become biased by the shorter (in hops) connection's throughput. The results for 6-hop, 9-hop and 12-hop TCP connections are shown in Figure 10. The bytes received by TCP receiver is compared with the bytes sent by the TCP sender.

From the results it is evident that the fraction of packets delivered successfully decreases with the increase in packet size. This is an expected result. Increasing the packet size increases the throughput till a certain threshold. For a short (in hops) connection this threshold occurs around 1000 bytes. Beyond this threshold, the gains of sending a larger packet are largely offset by the increase in false link failures. As we increase the hop-length of a TCP connection, any transmission potentially faces more interference. Thus, increasing the length of the connection will increase the probability of link failures at MAC layer.

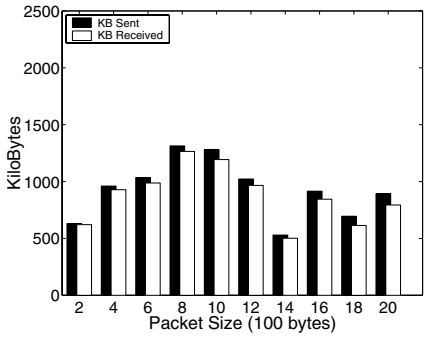
We observe that for the longer connections (9 and 12 hops) there is a decrease in throughput after the threshold. This largely due to the increase in false link failures cause by increased interference. The link failure and the packet loss reduce the TCP window size, which in turn reduces the overall throughput. This decrease cannot be compensated by the corresponding throughput increase due to larger packet size. We also observe that for longer path lengths, beyond the threshold there is an oscillatory behavior in TCP throughput. This is due to the fact that for longer paths the link failures occur randomly. These random link failures and the associated packet drops stifle the TCP sender's congestion window. Thus, we do not see a steadily decreasing trend in TCP throughput with packet size beyond the threshold.

During the transmission of a 1000-byte data packet a node within the interference range may receive 2 to 3 RTS packets. This follows from the default values for the slot duration, contention window used by binary exponential algorithm and the data transmission rate at the physical layer specified in the IEEE 802.11 standards. The node may not be able to respond to any of these RTS packets. If such a node cannot obtain the channel for the duration of two successive data packet transmissions by other nodes, it will still not infer a link failure as the number of retransmission

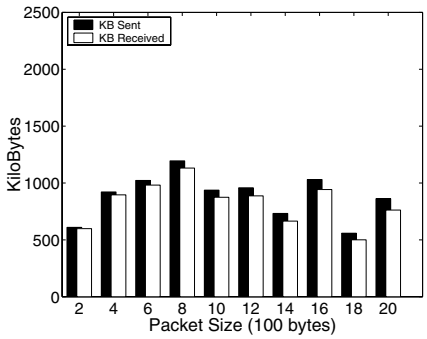
Proceedings of the DARPA Information Survivability Conference and Exposition (DISCEX'03)
0-7695-1897-4/03 $17.00 © 2003 IEEE



(a) 6-hop connection



(b) 9-hop connection



(c) 12-hop connection

Figure 10. Variations with Packet size.

attempts is below the retry limit of 7. However, when the packet size is 1400 bytes, a node within the interference range of a transmission may receive up to 4 RTS packets. In this case, medium capture by two successive packet transmissions will trigger a link failure. Thus, depending on the network topology and traffic conditions there is likely to be a threshold for TCP packet size, beyond which link failures due to interference become more probable. This also explains why increasing the retry limit in the IEEE 802.11 MAC protocol improves performance.

6. Conclusion

Our work focuses on TCP and IEEE 802.11 MAC parameters that affect performance in a multi-hop wireless environment. Our simulations and analysis show that both IEEE 802.11 MAC protocol and TCP contribute to the performance degradation when they are coupled.

The fundamental cause of degradation is the occurrence of false link failures at the MAC layer. An aggressive TCP sender causes an increased contention at the MAC layer which results in increased link failures. We show that the default value for the maximum TCP congestion window is too high. Our analysis shows that using a lower limit, which is a function of the number of hops in the TCP connection, provides better performance.

The IEEE 802.11 MAC protocol mandates a certain number of retransmission attempts for failed packets. Our simulations show that the default values suggested for these attempts are not optimal. The optimal values are a function of traffic load as well as mobility. For a static topology or a low mobility environment, increasing the retransmission limits results in significant improvement in performance. For a highly mobile environment, such an increase will result in increased delay in link failure detection.

We investigated the effects of the link failures when 2 TCP connections share an intermediate node. We observe that in such a case the throughput achieved by the TCP connection is primarily dependent on its hop-length relative to the other TCP connection. In the presence of a short hop-length connection, the longer connection does not get much bandwidth.

We observe the effect of TCP packet size on the overall throughput. We find that an increase in packet size reduces the fraction of packets delivered successfully. It also increases the interference related link failures and packet drops. However, larger packets provide better channel utilization. Thus, we find that there is a trade-off involved. We explore this trade-off and find that there is threshold packet size for TCP connections. Increasing packet size beyond this threshold results in degradation in throughput. The threshold is dependent on the path length as interference increases with number of hops.

Proceedings of the DARPA Information Survivability Conference and Exposition (DISCEX'03)
0-7695-1897-4/03 $17.00 © 2003 IEEE

7. Future work

Our work indicate that further work is necessary to make TCP and IEEE 802.11 MAC protocols compatible in a multi-hop environment. For the MAC protocol, timers and queue scheduling, and their interplay with TCP's back off schemes need further investigation. For TCP, congestion control mechanism needs more refinement when coupled with IEEE 802.11 MAC in a multi-hop wireless network.

8. Acknowledgements

This work was supported by the Fault-Tolerant Networks program of DARPA under contract F30602-01-2-0535, and by grants from Tata Consultancy Services and the DiMI program of the University of California.

References

[1] Bakre and B. R. Badrinath. I-TCP: Indirect TCP for mobile hosts. *Proc. 15th Int. Conf. Distributed Computing Systems*, 1995.

[2] B. Bakshi, P. Krishna, N. H. Vaidya, and D. K. Pradhan. Improving Performance of TCP over Wireless Networks. *Proc. Int. Conf. Distributed Computing Systems*, 1997.

[3] C. Barakat, E. Altman, and W. Dabbous. On TCP Performance in a Hetergeneous Network: A Survey. *IEEE Communications Magazine*, Jan 2000.

[4] B. Bensaou, Y. Wang, and C. C. Ko. Fair Media Access in 802.11 Based Wireless Ad-hoc Networks. *Proc. Mobihoc*, 2000.

[5] V. Bhargavan, A. Demers, S. Shenker, and L. Zhang. MACAW: A Media Access Protocol for Wireless LANs. *Proc. ACM SIGCOMM*, 1994.

[6] K. Brown and S. Singh. M-TCP: TCP for mobile cellular networks. *ACM Compueer Communication Review*, 27(5), 1997.

[7] K. Chaandran, S. Raghunathan, S. Venkatesan, and R. Prakash. A Feedback based Scheme for Improving of TCP Performance in Ad hoc Wireless Networks. *IEEE Personal Communications Magazine*, Feb 2001.

[8] K. Fall and K. Varadhan. notes and documentation, LBNL. *http://www.mash.cs.berkeley.edu/ns*, Aug 1998.

[9] M. Gerla, K. Tang, and R. Bagrodia. TCP Performance in Wireless Multihop Networks. *IEEE WMCSA*, 1999.

[10] G. Holland and N. Vaidya. Analysis of TCP performance over mobile ad hoc networks. *Proc. ACM Mobicom*, 1999.

[11] G. Holland, N. Vaidya, and P. Bahl. A Rate-Adaptive MAC Protocol for Multi-Hop Wireless Networks. *Proc. ACM Mobicom*, 2001.

[12] IEEE Std. 802.11. Wireless LAN Media Access Control (MAC) and Physical Layer (PHY) Specifications. 1999.

[13] P. Karn. MACA - A New Channel Access Method for Packet Radio. *Proc. 9th ARRL/CRRL Amateur Radio Computer Networking Conference*, Sept 1990.

[14] NS2. Network simulator. *http://www.isi.edu/nsnam*.

[15] C. Parsa and J. J. Garcia-Luna-Aceves. TULIP: A Link-Level Protocol for Improving TCP over Wireless Links. *Proc. IEEE WCNC*, 1999.

[16] C. E. Perkins, E. M. Royer, and S. R. Das. Ad Hoc On-Demand Distance-Vector (AODV) Routing. *IETF Internet draft (draft-ietf-manet-aodv-o6.txt)*.

[17] R. Rozovsky and P. R. Kumar. SEEDEX: A MAC Protocol for Ad hoc Networks. *Proc. ACM MobiHOC*, 2001.

[18] K. Tang and M. Gerla. Fair Sharing of MAC under TCP in Wireless Ad-hoc Networks. *Proc. IEEE MMT*, 1999.

[19] K. Xu, S. Bae, S. Lee, and M. Gerla. TCP behavior across multihop wireless networks and the wired internet. *Proc. WoWMoM*, 2002.

[20] S. Xu and T. Saadawi. Does the IEEE 802.11 MAC protocol Work Well in Multihop Wireless ad hoc networks. *IEEE Communications Magazine*, June 2001.

Proceedings of the DARPA Information Survivability Conference and Exposition (DISCEX'03)
0-7695-1897-4/03 $17.00 © 2003 IEEE