Verification of Common 802.11 MAC Model Assumptions

David Malone, Ian Dangerfield, and Doug Leith

Hamilton Institute, NUI Maynooth, Ireland

Abstract. There has been considerable success in analytic modeling of the 802.11 MAC layer. These models are based on a number of fundamental assumptions. In this paper we attempt to verify these assumptions by taking careful measurements using an 802.11e testbed with commodity hardware. We show that the assumptions do not always hold but our measurements offer insight as to why the models may still produce good predictions. To our knowledge, this is the first in-detail attempt to compare 802.11 models and their assumptions with experimental measurements from an 802.11 testbed. The measurements collect also allow us to test if the basic MAC operation adhere to the 802.11 standards.

1 Introduction

The analysis of the 802.11 CSMA/CA contention mechanism has generated a considerable literature. Two particularly successful lines of enquiry are the use of pure p-persistent modeling (e.g. [3]) and the per-station Markov chain technique (e.g. [2]). Modeling usually involves some assumptions, and in this respect models of 802.11 are no different. Both these models assume that transmission opportunities occur at a set of discrete times. These discrete times correspond to the contention counter decrements of the stations, equivalent to state transitions in the models, and result in an effective slotting of time. Note that this slotting based on MAC state transitions is different from the time slotting used by the PHY. A second assumption of these models is that to a station observing the wireless medium, every slot is equally likely to herald the beginning of a transmission by one or more other stations. In the models this usually manifests itself as a constant transmission or collision probability.

In this paper we will show detailed measurements collected from an experimental testbed to study these assumptions. This is with a view to understanding the nature of the predictive power of these models and to inform future modeling efforts. The contribution of this paper includes the first published measurements of packet collision probabilities from an experimental testbed and their comparison with model predictions and the first detailed comparison of measured and predicted throughputs over a range of conditions.

We are not the first to consider the impact of model assumptions. In particular, the modeling of 802.11e has required the special treatment of slots immediately
after a transmission in order to accommodate differentiation based on AIFS (e.g. \[1,9,11,6,4\]). In [13] the nonuniform nature of slots is used to motivate an 802.11e model that moves away from these assumptions.

2 Test Bed Setup

The 802.11e wireless testbed is configured in infrastructure mode. It consists of a desktop PC acting as an access point, 18 PC-based embedded Linux boxes based on the Soekris net4801 [7] and one desktop PC acting as client stations. The PC acting as a client records delay measurements and retry attempts for each of its packets, but otherwise behaves as an ordinary client station. All systems are equipped with an Atheros AR5215 802.11b/g PCI card with an external antenna. All stations, including the AP, use a Linux 2.6.8.1 kernel and a version of the MADWiFi [8] wireless driver modified to allow us to adjust the 802.11e CWmin, AIFS and TXOP parameters. All of the systems are also equipped with a 100Mbps wired Ethernet port, which is used for control of the testbed from a PC. Specific vendor features on the wireless card, such as turbo mode, are disabled. All of the tests are performed using the 802.11b physical maximal data transmission rate of 11Mbps with RTS/CTS disabled and the channel number explicitly set. Since the wireless stations are based on low power embedded systems, we have tested these wireless stations to confirm that the hardware performance (especially the CPU) is not a bottleneck for wireless transmissions at the 11Mbps PHY rate used. As noted above, a desktop PC is used as a client to record the per-packet measurements, including numbers of retries and MAC-level service time. A PC is used to ensure that there is ample disk space, RAM and CPU resources available so that collection of statistics not impact on the transmission of packets.

Several software tools are used within the testbed to generate network traffic and collect performance measurements. To generate wireless network traffic we use mgen. We will often use Poisson traffic, as many of the analytic models make independent or Markov assumptions about the system being analysed. While many different network monitoring programs and wireless sniffers exist, no single tool provides all of the functionality required and so we have used a number of common tools including tcpdump. Network management and control of traffic sources is carried out using ssh over the wired network.

3 Collision Probability and Packet Timing Measurement

Our testbed makes used of standard commodity hardware. In [5] we developed a measurement technique that only uses the clock on the sender, to avoid the need for synchronisation. By requesting an interrupt after each successful transmission we can determine the time that the ACK has been received. We may also record the time that the packet was added to the hardware queue, and by inverting the standard FIFO queueing recursion we can determine the time the MAC spent processing the packet. This process is illustrated in Figure [1] For