Experimental Evaluation of 802.11e EDCA for Enhanced Voice over WLAN Performance

Ian Dangerfield, David Malone, Douglas J. Leith

Abstract— **The 802.11e MAC protocol extends the 802.11 CSMA/CA contention mechanism by allowing the adjustment of MAC parameters that were previously fixed. While the 802.11e protocol has been extensively studied, this work is almost entirely confined to analytical and simulation studies. In this paper we demonstrate a technique for measuring one-way delay in an 802.11e hardware testbed and thereby study delay in the context of protecting a voice call competing against data traffic. We demonstrate that with the standard 802.11b MAC settings greedy data traffic is able to seize bandwidth from a low-rate voice call. Only 5 competing data stations are needed in order to induce a voice call loss rate exceeding 10%, which in practice would lead to an unacceptable level of voice quality and dropping of the call. We present experimental measurements which demonstrate that the use of 802.11e to provide a practical solution that can successfully deliver quality of service to voice traffic in a mixed voice/data environment. To our knowledge, this is the first experimental demonstration of accurate one way delay measurements being used to show the priorisation of voice in an 802.11e hardware test-bed.**

Keywords— **802.11e, CSMA/CA, test-bed, voice, measurement**

I. Introduction

The new 802.11e MAC protocol [1] extends the standard 802.11 CSMA/CA contention mechanism by allowing the adjustment of MAC parameters that were previously fixed. While the 802.11e protocol has been extensively studied in the literature, this work is almost entirely confined to analytical and simulation studies. Owing to the lack of available hardware, there have been very few experimental studies evaluating the performance of the new 802.11e protocol. Hardware is, however, now available which allows us to investigate 802.11e EDCA operation in a real testing environment. We have constructed an 802.11e hardware testbed network and in this paper our aim is make use of this testbed to perform experimental measurement and validation of 802.11e operation.

As a first step, in [3] we compared our expectations (from theory and simulation) with the behaviour of an actual 802.11e implementation. This allows us to identify the limitations of such predictions. There is an extensive literature containing simulation [2], [4] and analytic [5], [6] studies/comparisons of the 802.11 and 802.11e MAC mechanisms. A number of experimental studies, albeit in the context of 802.11 rather than 802.11e, also suggest that there may exist some gap between theoretical predictions and practical performance [7], [8].

In the present paper we investigate the impact of competing data traffic on the quality of service received by a voice call in an 802.11e WLAN. It is readily demonstrated that data traffic adversely affects a voice call in an 802.11b WLAN. Figure 1 shows the measured throughput of a 64Kbs voice call as the number of competing stations is varied — these are experimental measurements taken using our WLAN testbed described below. With the standard 802.11b MAC it can be seen that data traffic is able to seize bandwidth from the low-rate voice call. Only 5 competing data stations are needed in order to induce a voice call loss rate exceeding 10%, which in practice would lead to an unacceptable level of voice quality and dropping of the call. With the ongoing roll-out of VoIP and the widespread trend towards wireless connectivity at the network edge, there is a real need for improved quality of service for VoIP over WLAN. In this paper we present experimental measurements to demonstrate that the flexibility of 802.11e provides practical solutions that can successfully deliver quality of service to voice traffic in a mixed voice/data environment.

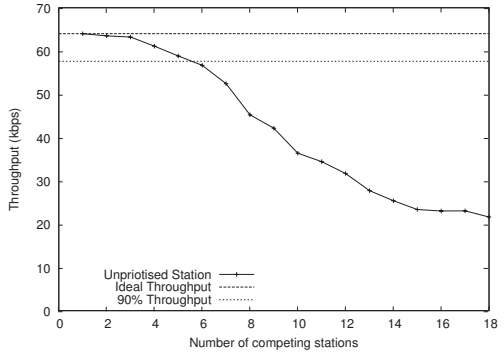

Fig. 1. Throughput for a G.711-like voice call in an 802.11b infrastructure WLAN as the number of competing data stations is varied. See Table I for details. Data stations always have a packet to send with 1470 byte payload.

Networks with mixed voice/data traffic have previously been considered in [9] where an experimental study of the capacity of voice in an 802.11b network is performed. In [10] the implementation and validation of a priority queue scheme at the driver level above an 802.11b MAC is considered. Some work has considered the voice call capacity of 802.11 networks rather than adjustment the MAC layer behaviour itself. For example, in [11], a back-of-envelope calculation for maximum capacity of a WLAN is presented and shown to be a useful estimate. The authors also consider, using simulation, how delay constraints and bit error rates impact the capacity of the network. Other metrics

Work supported by Science Foundation Ireland grant IN3/03/I346.
The authors are with the Hamilton Institute, National University of Ireland, Maynooth, Co. Kildare, Ireland.

0-7803-9550-6/06/$20.00 © 2006 IEEE

for voice capacity are also used in, for example, [9], [12].

Our focus here is on the experimental measurement of the performance of proposed 802.11e solutions. We present a practical technique for measuring the one-way delay between 802.11 MAC layers and combine this technique with the tuning of 802.11e MAC parameters to show how Voice over WLAN can be protected from competing data transfers. To our knowledge, this is the first test-bed measurements of the priorisation of voice using 802.11e.

II. 802.11e EDCA Summary

In this section we will briefly outline the relevant parts of the 802.11 DCF and 802.11e EDCA MAC layers. The 802.11 MAC layer CSMA/CA mechanism uses a binary exponential back-off algorithm to regulate access to the shared wireless channel. On detecting the wireless medium to be idle for a period DIFS, each station initialises a counter to a random number selected uniformly up to CW. Time is slotted and this counter is decremented once during each slot that the medium is observed idle. A significant feature is that the countdown halts when the medium becomes busy and resumes after the medium is idle again for a period DIFS. Once the counter reaches zero the station attempts transmission and can transmit for a duration up to a maximum time TXOP (defined to be one packet in 802.11a/b/g). If two or more stations attempt to transmit simultaneously, a collision occurs. Colliding stations double their CW (up to a maximum value, CWmax), select a new back-off counter uniformly and the process repeats. After successful transmission, CW is reset to its minimal value CWmin and a new countdown starts regardless of the presence of a packet at the MAC. If a packet arrives at the MAC after the countdown is completed, the station senses the medium. If the medium is idle, the station attempts transmission immediately; if it is busy, another back-off counter is chosen from the minimum interval. The new 802.11e MAC enables the values of DIFS (called AIFS in 802.11e), CWmin, CWmax and TXOP to be set on a per-class basis for each station. That is, traffic is directed to up to four different queues at each station, with each queue assigned different MAC parameter values.

We will vary the parameters AIFS, CWmin and TXOP in this paper. AIFS is adjustable in units of the 802.11 slot length and we say AIFS 0 to mean DIFS, AIFS 1 to mean DIFS plus one slot and so on. CWmin is adjustable in powers of two i.e. as 2^k with integer k. TXOP is a length of time, specified in microseconds.

III. Testbed Setup

The 802.11e wireless testbed is configured in infrastructure mode. It consists of a desktop PC acting as an access point (AP), 18 PC-based embedded Linux boxes based on the Soekris net4801 [13] and one desktop PC acting as client stations. The PC acting as a client records delay measurements for each of its packets, but otherwise behaves as an ordinary client station. All systems are equipped with an Atheros 802.11b/g PCI card with an external antenna. The system hardware configuration is summarised in Table I. All nodes, including the AP, use a Linux 2.6.8.1 kernel and a version of the MADWiFi [14] 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 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 nodes 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 delay measurements. This is 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.

Hardware	model	spec
1× AP	Dell GX 260	2.66Ghz P4
18× node	Soekris net4801	266Mhz 586
1× measurement node	Dell GX 270	2.8Ghz P4
WLAN NIC	D-Link DWL-G520	Atheros AR5212

TABLE I
Testbed Hardware Summary

parameter	default	used
interface tx queue	199 packets	10 packets
driver tx queue	200 packets	2 packets
MAC Preamble	short	long
MAC Data rate	54Mbps	11Mbps
MAC Retries	11	11

TABLE II
Testbed Parameters Summary

The configuration of the various network buffers and MAC parameters is detailed in Table II. We have shortened certain queues to reduce buffering between our tests and the network (eg. in the case of TCP). We do not make significant use of this feature in this paper, but retain the buffer sizing for consistency with [3].

The testbed was calibrated as described in [3] by adjusting the positions of stations and antennae until the throughputs achieved by all stations were roughly similar.

Several software tools are used within the testbed to generate network traffic and collect performance measurements. To generate wireless network traffic and to measure throughput we use mgen[15]. 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[16]. Network management and control of traffic sources is carried out using ssh over the wired network.

IV. Measurement of Delay

A key feature that distinguishes voice traffic from data traffic is its sensitivity to network delay. In the present context we are interested, in particular, in the network delay associated with winning access to transmission opportunities in an 802.11 WLAN. This MAC access delay is associated with the contention mechanism used in 802.11 WLANs. The MAC layer delay, i.e. the delay from a packet becoming eligible for transmission (reaching the head of the hardware interface queue) to final successful transmission, can range from a few hundred microseconds to hundreds of milliseconds, depending on network conditions.

Synchronising the clocks on a sender/receiver to within a few hundred microseconds is not practical with a standard protocol, such as NTP[17]. It is possible that specialised protocols for measurement like [18] could be used. We considered using events simultaneously observable at the sender and the receiver, such as reception of a wireless broadcast packet, to correct for clock skew. Using this technique we were able to observe NTP adjusting clock frequencies. For this technique we had to request a hardware interrupt for every packet received, in order to prevent packets being queued for different times on different stations. While this technique looked promising, we found that variations in interrupt processing time at the sender and receiver, combined with the need to interpolate between broadcast packets, made accurate measurements difficult.

The measurement technique that was finally used was to only use the clock on the sender, to avoid the need for synchronisation. This can be done because the successful transmission of an 802.11 packet is indicated by the sender receiving a fixed length MAC-level ACK from the station that successfully received the packet. This ACK does not have to contend for access to the medium and its transmission begins $10\mu s$ after the successful transmission ends.

Thus, by requesting an interrupt after each successful transmission[1], 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 2. Note, this technique can be generalised to get the transmission time in other cases, such as when the RTC/CTS mechanism is in use, as long as the time of interest can be derived from the MAC service time.

V. Validation

As a basic validation of our technique, we measured the time to transmit packets of various sizes on a unloaded 802.11b WLAN. In this case, there should be no stochastic back-off and minimal failed packet retransmissions due to collisions. In practice noise and beacon frames may actually cause occasional retransmission, so we use the median

[1]If a packet is not successfully transmitted because the MAC retry limit is exceeded, we would also receive an interrupt. However, very few such events are seen in our experiments.

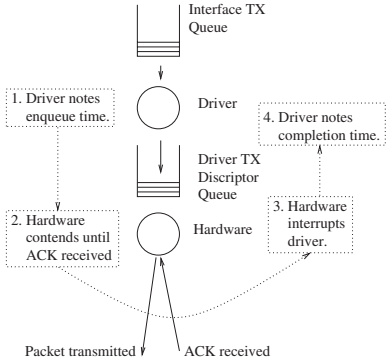

Fig. 2. Schematic of delay measurement technique.

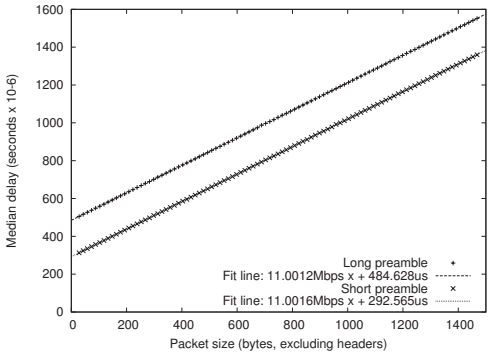

Fig. 3. Transmission time against IP packet size on a quiet network.

value observed. Effectively the packet transmission time should correspond to sending PHY headers, MAC headers, data, a short interframe space and then an ACK.

Figure 3 shows the results, where we see the transmission time is a linear function of the the packet size, as expected. Fitting a line to these points we find the slope is very close to 11Mbps. By setting the driver to use a long preamble ($192\mu s$ rather than $96\mu s$) we see the transmission time increase by $192\mu s$ (to the nearest microsecond), as the overhead is added to both the data packet and the ACK. The value of the intercept should be the time to transmit a packet with no payload. For the long preamble we expect this to be,

$$DIFS + PLCP + MAC + CRC + SIFS + PLCP + ACK =$$
$$50 + 192 + 24 * 8/11 + 4 * 8/11 + 10 + 192 + 14 * 8/11 = 474.545\mu s$$

so it can be seen that our measured values seem to be within 10 μs of the expected values. Note that though the short preamble times are significantly shorter, the absolute accuracy seems similar.

Let us now move to the 802.11e parameters. In Figure 4 we show two graphs representing the impact of TXOP on two competing stations. Both stations are saturated (always have a packet to send), so back-off associated with contention will be present in our measurements. In these experiments, one station has its TXOP fixed so that it can only transmit one packet at each transmission opportunity. We vary the TXOP of the other and show the

achieved throughput of both stations and the throughput of the system as a whole in the graph on the left. We see the expected steps as the TXOP crosses a packet boundary[2]. Note also that the overall system throughput increases as we increase TXOP, because the overhead of contention is being amortised over several packets.

On the right of Figure 4, we show both the measured mean delay for the station with the variable TXOP and the product of the mean delay with the measured throughput. For saturated stations this product should be the packet size. As expected, we see the saturated station's mean delay decreasing, demonstrating the amortisation of overhead, while the throughput-delay product remains approximately constant at about 1470 bytes.

Figure 5 shows a similar experiment where we hold CWmin fixed at 31 on one station, while we vary CWmin (in powers of two) on the other station. We see that a doubling of CWmin, corresponding to a doubling of how many slots the station must count down for on average, results in an approximate doubling of the mean packet delay. The relative throughputs are also approximately in proportion to the ratio of the CWmin values.

Note that the overall throughput drops as we increase CWmin. This is because the optimal CWmin for two saturated stations is small, much less than 15, the smallest value we consider. However, the drop is quite small (roughly 10% over the range of CWmin that we plot).

Figure 6 shows how AIFS changes throughput and delay. It is expected that the effect of AIFS will be load dependent, and we see that for two competing nodes that AIFS has a smaller impact than shifting CWmin, in terms of separation of throughput and delay. Also, it causes a slight decrease in total system throughput, as one station must now wait longer to access the medium.

We have shown that the delay and throughput measurements have the expected relationship. In [3] we show how throughput predictions are in line with analytic or simulation results, thus, due to lack of space we do not present that comparison here.

Now that we have validated our technique for measuring delays and shown how the 802.11e EDCA parameters affect delay in some simple situations, we will move to a the more challenging task of prioritising a voice call.

VI. Prioritising voice

The scheme we use to prioritise voice is a simple one: we increase the AIFS value used by other competing stations[3]. Increasing a station's AIFS value results in an increased delay after *every* transmission on the network before that station can continue decrementing its counters. As the effect of AIFS therefore becomes stronger as the network's load increases we expect that as load increases a fixed value of AIFS may well be sufficient to achieve QoS targets for voice.

[2]Note, even though we have set the buffer between the driver and the hardware to be 2 packets, we can still use TXOP to send more than 2 packets as the driver can add a new packet to the hardware queue while another packet is being transmitted.

[3]It would be preferable to reduce AIFS for the voice rather than increasing AIFS for the competing stations, however this is out of scope of the standard, as it can interfere with the transmission of management frames and ACKs.

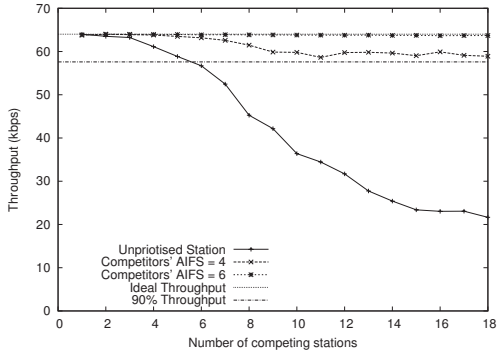

Fig. 7. Throughput by a voice call competing with saturated stations.

We consider voice as simple 64Kbps stream with packets every 10ms. Each packet has a payload of 80 bytes. This voice call shares the network with a number of stations that are saturated, transmitting a 1470 byte packet whenever the MAC allows. Each experiment is run for 20 minutes for smaller numbers of stations and 30 minutes for larger numbers of stations. This is because when unprioritised, the throughput of the voice call falls rapidly, and we must run the experiment for longer to transmit enough packets to accumulate accurate delay statistics. Figure 7 shows the throughput of an unprioritised station falling. Figure 8 shows how at the same time delays increase. Note that when unprioritised the delay continues to increase significantly as more competing stations are added.

Figure 7 and Figure 8 also show the throughput and delay when the competing stations have AIFS values of 4 and 6. Note that in contrast to the unprioritised case, the delay begins to level off at about 8–10 stations and throughput stabilises. This seems to confirm that a fixed AIFS value will be sufficient to prioritise a voice call against quite large numbers of competing stations.

An AIFS value of 4 can be seen to keep the mean delay just below the inter-packet time, which is required for the queue to be stable. It also keeps the throughput above 90%. An AIFS of 6 keeps the mean delay well below the inter-packet time and achieves full throughput.

However, not just mean delays are important. Figure 9, 10 and 11 show how the transmission times are distributed in when the call is unprioritised, prioritised with AIFS 4 and AIFS 6 respectively. These graphs show regular plateaux corresponding to the number of packets transmitted before each voice packet is successfully transmitted.

In the unprioritised case, we can see that with 12 competing stations, less than half of the voice packet can be cleared before the next packet arrives. For the prioritised case, we can see that the delay distribution varies less as we consider larger numbers of competing nodes. Almost 70% of packets can be cleared before the next arrives with an AIFS of 4 and for an AIFS of 6 this is almost 90%.

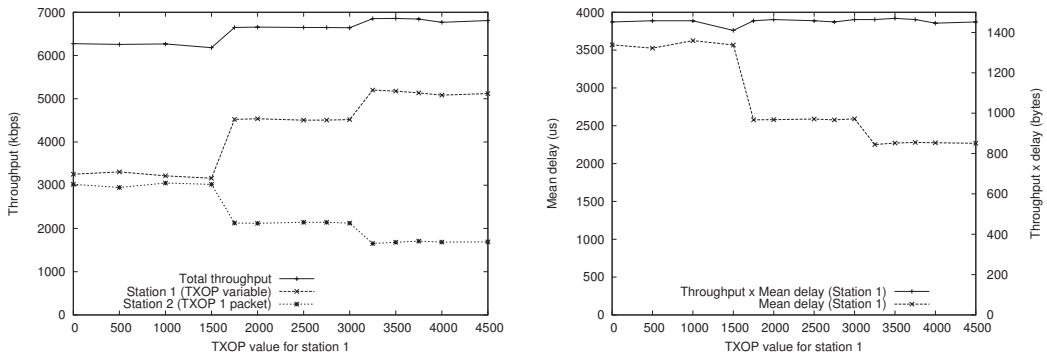

Fig. 4. The impact of TXOP on two competing saturated stations.

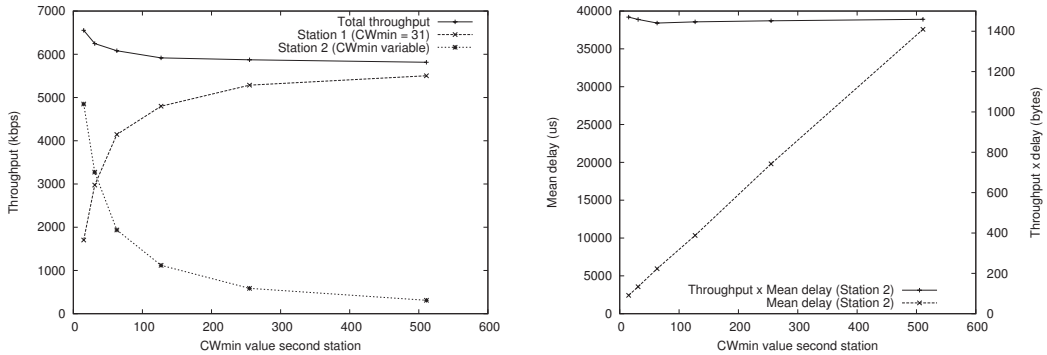

Fig. 5. The impact of CWmin on two competing saturated stations.

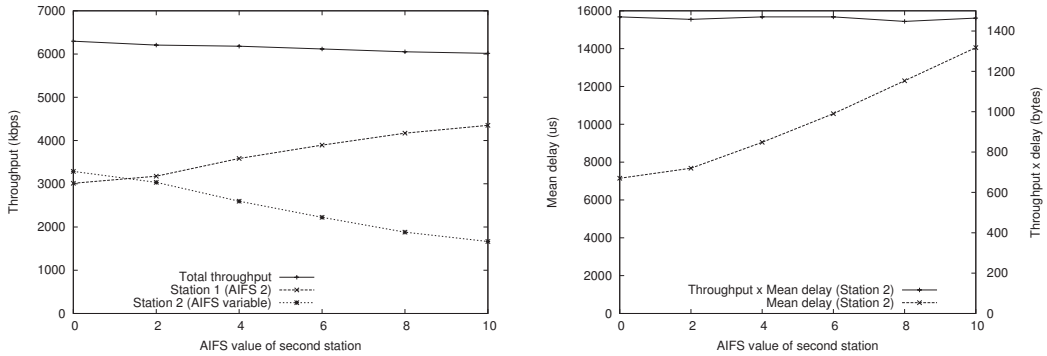

Fig. 6. The impact of AIFS on two competing saturated stations.

One concern is that packet delays may not be independent of one another. Correlated delays might lead to extra queueing delay or packet loss. The construction of the MAC suggests that they should be independent, however factors such as fading and noise might introduce correlations. Figure 12 shows the autocorrelation for the sequence of observed packet delays. We can see that there is practically no correlation between packet delays, suggesting that techniques for G/GI/1 queues may be appropriate.

Recall, that in Section V we plotted the product of the throughput by the mean delay, which for a saturated system gives the packet size. In Figure 13 we show the product of throughput by delay, scaled so that the packet size is 1. This represents the proportion of time that the MAC layer handling the voice call has a packet to transmit.

We can see that the unprioritised station becomes saturated when there are 8–9 competing stations. Both AIFS 4 and AIFS 6 prevent the MAC layer becoming saturated.

VII. Conclusion

We have devised a technique for measuring the MAC level delay in 802.11 networks using per-packet interrupts. We have demonstrated the accuracy of this method in our test-bed and shown it to be highly accurate, capable of achieving accuracies of tens of microseconds. We have shown how the 802.11e parameters AIFS, CWmin and TXOP change the delay experienced by a station.

Using these techniques, we have studied delay in the con-

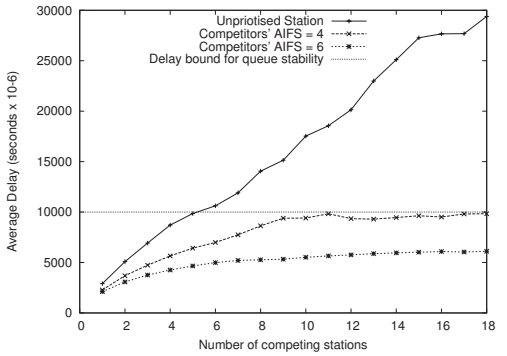

Fig. 8. Mean delay for a voice call competing with saturated stations.

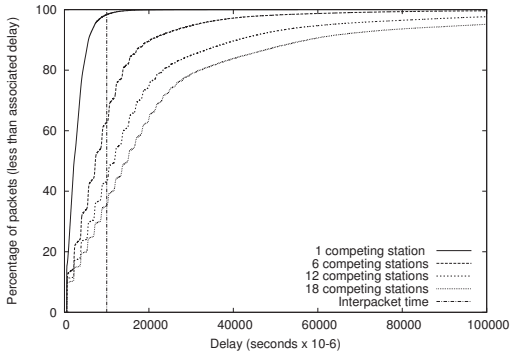

Fig. 9. CDF for the delay of an unprioritised voice call.

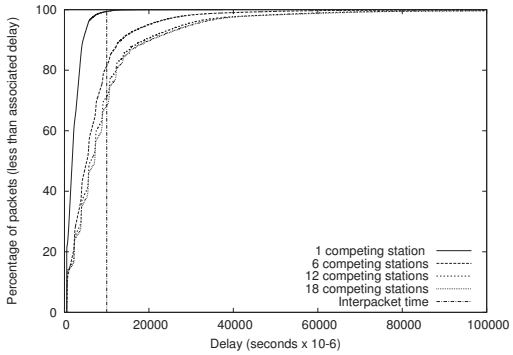

Fig. 10. CDF for the delay of a prioritised voice call (AIFS 4).

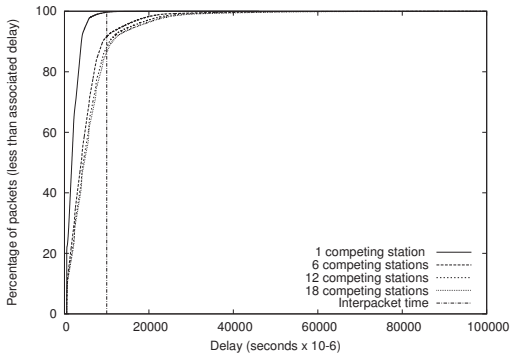

Fig. 11. CDF for the delay of a prioritised voice call (AIFS 6).

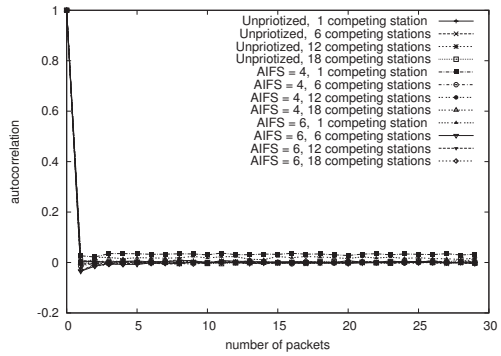

Fig. 12. Autocorrelation for sequence of packet delays.

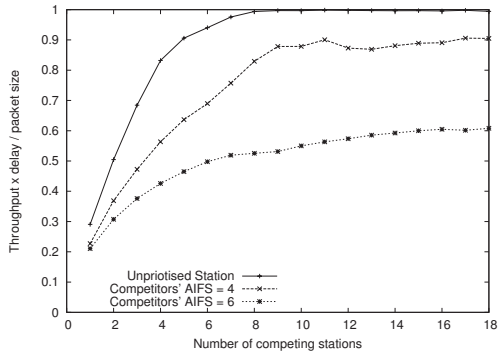

Fig. 13. Proportion of time the MAC layer is busy.

text of protecting a voice call competing against data traffic in an 802.11 infrastructure mode network. We demonstrate that with the standard 802.11b MAC settings greedy data traffic is able to seize bandwidth from a low-rate voice call. Only 5 competing data stations are needed in order to induce a voice call loss rate exceeding 10%, which in practice would lead to an unacceptable level of voice quality and dropping of the call. Using the flexibility provided by the new 802.11e MAC, we demonstrate that modest values of AIFS can be used to protect a voice call against large numbers of data stations, maintaining throughput, mean delays and delay distributions in a range where high voice call quality can be expected. Our results also indicate that the per-packet delays show little correlation, thus techniques for studying queues with independent service times may be useful in analysing performance.

To our knowledge, this is the first experimental demonstration of accurate one way delay measurements being used to show the prioritisation of voice in an 802.11e hardware test-bed.

References

[1] IEEE P802.11E/D9.0 Draft Standard, August 2004.
[2] Q. Ni, "ns-802.11e EDCF for IEEE 802.11e Wireless LAN", `http://www-sop.inria.fr/planete/qni/`, November 2002.
[3] A. Ng, D. Malone and D.J. Leith, "Experimental Evaluation of TCP Performance and Fairness in an 802.11e Test-bed", *ACM SIGCOMM E-WIND workshop*, August 2005.
[4] S. Wiethölter and C. Hoene, "Design and verification of an IEEE

802.11e EDCF simulation model in ns-2.26," Technische Universität Berlin, Tech. Rep. TKN-03-019, November 2003.
[5] G. Bianchi, "Performance analysis of IEEE 802.11 distributed coordination function," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 3, pp. 535–547, March 2000.
[6] R. Battiti, B. Li, "Supporting service differentiation with enhancements of the IEEE 802.11 MAC protocol: models and analysis", Technical Report DIT-03-024, University of Trento, May 2003.
[7] E. Pelletta, H. Velayos, "Performance measurements of the saturation throughput in IEEE 802.11 access points", WiOpt, April 2005.
[8] M. Franceschinis, M. Mellia, M. Meo, M. Munafo, "Measuring TCP over WiFi: A Real Case", WiNMee 2005, April 2005.
[9] F. Anjum, M. Elaoud, D. Famolari, A. Ghosh, R. Vaidyanathan, A. Dutta, P. Agrawal, T. Kodama and Y. Katsube, "Voice performance in WLAN networks — an experimental study", IEEE GLOBECOM 2003.
[10] J. Yu, S. Choi and J. Lee, "Enhancement of VoIP over IEEE 802.11 WLAN via Dual Queue Strategy", International Conference on Communications, 2004.
[11] D.P. Hole and F.A. Tobagi, "Capacity of an IEEE 802.11b Wireless LAN Supporting VoIP", International Conference on Communications, 2004.
[12] M. Coupechoux, V. Kumar and L. Brignol, "Voice over IEEE 802.11b capacity", 16th ITC Specialist Seminar on Performance Evaluation of Wireless and Mobile Networks, 2004".
[13] Soekris Engineering, `http://www.soekris.com/`.
[14] Multiband Atheros Driver for WiFi (MADWiFi), `http://sourceforge.net/projects/madwifi/`, October 2005 version.
[15] MGEN, The Multi-Generator Toolset, `http://mgen.pf.itd.nrl.navy.mil/`.
[16] tcpdump, `http://www.tcpdump.org/`.
[17] D. Mills et al, "NTP: The Network Time Protocol", `http://www.ntp.org/`.
[18] D. Veitch, S. Babu and A. Pàsztor, "Robust Synchronization of Software Clocks Across the Internet", ACM SIGCOMM Internet Measurement Conference, October 2004.