

Experiments for Real-Time Communication Contracts in IEEE 802.11e EDCA Networks

Michal Sojka, Martin Molnár, Zdeněk Hanzálek*

{sojka, molnam, hanzalek}@fel.cvut.cz

Czech Technical University in Prague, Faculty of Electrical Engineering
Technická 2, 166 27 Praha, Czech Republic

Abstract

In this paper we describe basic experiments measuring communication delays in IEEE 802.11e EDCA networks. Based on results of these experiments we have designed a simple utilization based admission test to support wireless networks in a contract framework.

Keywords: 802.11e, EDCA, measurement, latency, admission test

1. Introduction

Developing complex real-time applications is not an easy task because it requires engineers not only to create the application, but also to prove its timing correctness. To certify that the application fulfils its timing requirements it is necessary to perform a schedulability analysis, which can be difficult for complex and dynamic applications. For distributed applications, the analysis is even harder because timing is influenced by both processing delays and communication delays [1]. Moreover, many engineers are unfamiliar with the appropriate modeling and analysis techniques. To solve the problem of performing schedulability analysis one can develop his applications under a *contract framework*, where the application developer writes the application code and specifies its timing requirements (*contracts*). The framework provides on-line admission tests, which automatically compute the schedulability analysis to determine whether the timing requirements of a new application can be fulfilled without violating timing of already running applications. In case of negative result, the new applications are not allowed to run.

The contract framework, which is being developed within FRESOR project [2], will contain admission tests for different types of resources (CPU, network, disk, ...) with the goal of providing timing analysis of the whole system. In order for FRESOR to support applications that communicate through IEEE 802.11e based wireless networks (WLANs), we are developing an admission test for this resource. In this paper we describe our initial experiments, some design ideas and a simple admission test.

*This work was supported by the Ministry of Education of the Czech Republic under project 1M0567 (CAK) and by the FP6/2005/IST/5-034026 European Commission Project named FRESOR.

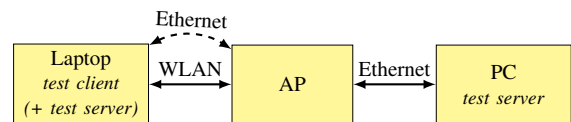


Figure 1. Our testbed setup

For space reasons, the reader is expected to be familiar with IEEE 802.11e standard [3].

2. Used technologies

We have decided to build the FRESOR wireless network support upon the Enhanced Distributed Channel Access (EDCA) technique even if HCF Controlled Channel Access (HCCA) technique would be more appropriate for real-time communication. The reason is that hardware supporting HCCA is very rare today. The development platform is Linux OS. In the recent Linux kernels, a wireless networking is unified by the new mac80211 wireless stack where access category (AC) of data packets can be specified by setting Type of Service (TOS) fields of IP datagrams.

2.1. Testbed setup

The experiments were conducted on a testbed (see Figure 1) where one station was a laptop running Linux 2.6.24 with Ovislink WMM-3000PCM Cardbus adapter containing Ralink RT2600 chip (`rt61pci` driver). The station was associated to an access point (AP) Linksys WRT54G ver. 7, which was connected by 100 Mbps Ethernet to a PC running Linux 2.6.22. This way we had two stations (Laptop and AP) competing for the wireless medium.

It is clear that with this setup the probability of collision is quite low and the measured results can be different if there are more stations. We will address this issue in our future work.

We have two testing applications. The *test server* serves as a loopback i.e. it listens for incoming UDP packets and sends them back to whoever who sent them, using the same TOS flags. The *test client* produces data streams of desired average bandwidth, packet size and access category, sends them to the *test server* and processes the responses. Both server and client add time-stamps to every packet payload so that the client can measure round-trip

time and (if time in server and client stations is synchronized) one-way delays. The scheduling policy of both *test client's* and *test server's* sending and receiving threads is set to SCHED_FIFO to minimize the influence of CPU scheduler to measured times. It was not necessary to use any real-time extensions to Linux kernel as we have used low baud rates and the latencies caused by the OS scheduler was several orders of magnitude below network latencies.

The parameters of EDCA queues were set to the default values defined in [3]. Transmission bit-rate was fixed to 1 Mbit/s on both sides to eliminate automatic rate control algorithms.

Since we were not able to synchronize the clock to within a few hundred microseconds using Network Time Protocol (NTP) and didn't want to use a more complicated synchronization techniques [4], for some experiments the setup was different. Both *test server* and *test client* were running in the same machine but the communication between WLAN and Ethernet interfaces were handled externally thanks to the *send-to-self* patch¹ for Linux kernel (see dashed line in Figure 1).

3. Experiments

The results of our experiments are presented in the graphs below in the form of cumulative histogram of delay. The horizontal axis represents the measured time (divided by two in the case of round-trip time) and the vertical axis the percentage of received packets with delay less or equal to the value on the horizontal axis. The exact parameters of streams generated by *test client* application are shown above every plot. The packet size values don't include any headers (UDP, IP, MAC). As the measured time includes the transmission time of the packet, we keep the packet size of all streams the same to eliminate the influence of different transmission time. Also, to excite all possible behavior of the stochastic system represented by IEEE 802.11 MAC layer, the delay between send attempts is an evenly distributed random number with mean value of desired period and maximal deviation equal to 50% of period. All experiments ran for 60 seconds.

3.1. Basic experiments

The experiment in Fig. 2 shows the delays of all access categories (voice, video, best-effort, background) under non-saturation condition. The worst-case delay of AC_VO was around 70 ms and the one of AC_BK was 240 ms.

In the second experiment (see Figure 3), we have slightly increased the bandwidth of all streams and AC_BK queue got into saturation. We can see that we have received only 10 packets per second instead of requested 17 and the worst-case delay increased to 5.4 seconds. As we show below, the major part of this delay is caused by waiting in transmission queues.

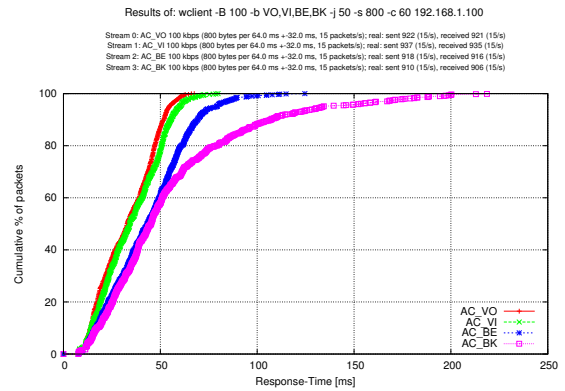


Figure 2. Delay of all access categories under non-saturation condition

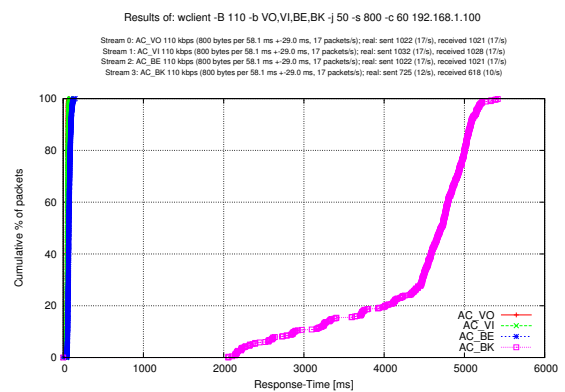


Figure 3. Delay of all access categories where AC_BK is under saturation

3.2. AC interdependencies

In these experiments we have measured the influence of AC_BE stream of different bandwidths to the delays of AC_VO and AC_VI streams. The bandwidth of AC_VO and AC_VI was fixed to 100 kbit/s and the bandwidth of AC_BE was changed from zero to 340 kbit/s.

Figure 4 shows a non-saturated case with 20 kbps AC_BE stream. All subsequent AC_BE bandwidths produced similar results until 200 kbps (Figure 5), where AC_BE bandwidth reached the saturation boundary and worst-case delay increased to 210 ms. If AC_BE bandwidth is slightly increased to 220 kbps (Figure 6), the worst-case delay increases rapidly to 2 seconds and remain there obviously even for higher requested bandwidths. The real AC_BE bandwidth did not exceed 200 kbps. The worst-case delay of AC_VO and AC_VI ranges from 30 ms in the non-saturated case, to 100 ms in all saturated cases.

The ramp between 50 and 1550 ms on the saturated (AC_BK) graph in Fig. 6 corresponds to the state where OS/HW queues are being filled. In the beginning, the queues are empty so that packets don't wait in queues and the delay is short. As the queues are filled more and more

¹<http://www.ssi.bg/~ja/#loop>

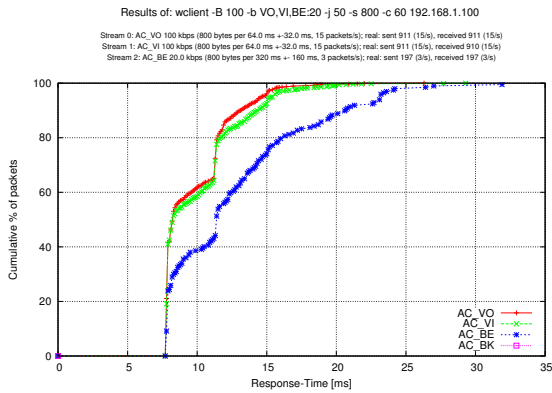


Figure 4. Influence of AC_BE at 20 kbps on AC_VO and AC_VI

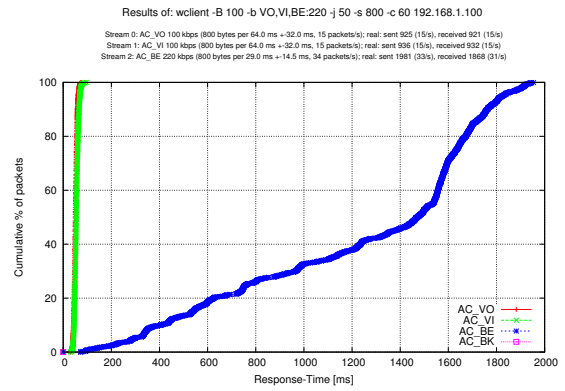


Figure 6. Influence of AC_BE at 220 kbps on AC_VO and AC_VI

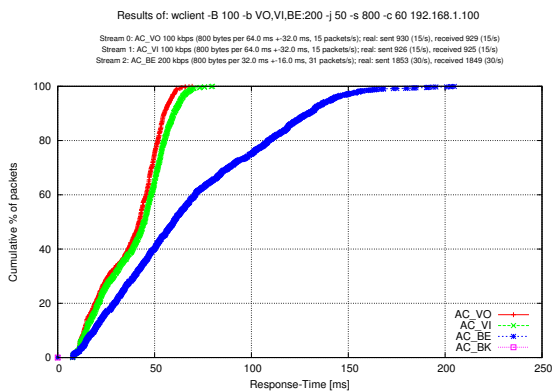


Figure 5. Influence of AC_BE at 200 kbps on AC_VO and AC_VI

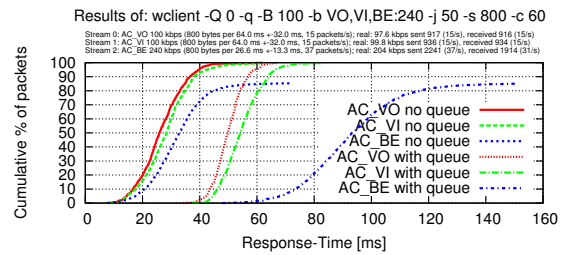


Figure 7. Influence of socket send queue size to delays. Two scenarios with SO_SNDBUF set to 0 and 3000.

the time spent in queues is longer and longer. Then there is notable turn at 1550 ms, which corresponds to the state where the queues are fully filled and packets starts to be dropped.

3.3. Influence of queue size; AP difference

When we decreased the maximum size of socket buffers (with `setsockopt` and `SO_SNDBUF`), we were able to decrease the maximum delay (see Figure 7). With zero byte buffers (the top graph), the lowest delay was achieved, but obviously when multiple threads use the same socket (in non-blocking mode) then the packet loss was higher (not depicted here).

Since parameters AIFSN, CWmin and CWmax can be different for AP and non-AP STAs [3], we have also evaluated how one-way delay depends on direction (non-AP to AP and vice versa). For this experiment we have used the testbed setup with both *test client* and *test server* in one station (dashed lines in Fig. 1). The results can be seen in Figure 8. The figure shows that the delays are shorter when the AP is the sender. This is obvious because AP must have precedence over non-AP stations or otherwise it would be overloaded by packets coming from stations.

3.4. Summary

From the results presented in this section we conclude that, in order to use IEEE 802.11e networks for real-time communication: (1) Saturation must be avoided for high priority ACs (VO, VI). (2) If saturation is not avoided for non-real-time (background) access categories, communication delay of real-time traffic increases (approximately by 100%). (3) Decreasing the number of socket buffers lowers delays but degrades performance.

4. Simple admission test

With respect to the summary in the previous section and before implementing quite difficult and computation demanding admission test based on [5], we have decided to start with a very simple and not much universal *utilization based* admission test whose goal is only to avoid saturation.

For each contract (stream) we determine how many UDP packets may be sent according to the negotiated budget B (bytes per period) and MTU^2 . For each packet we calculate its transmission time including all overheads: lower layer headers (UDP, IP, LLC, MAC), ACK packet and SIFS, PLCP preamble and PLCP header (and signal extension for ERP rates defined by IEEE 802.11g) for

²Maximum Transmission Unit

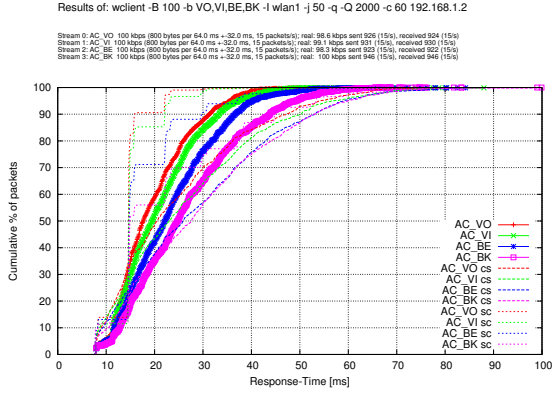


Figure 8. Difference between AP and non-AP transmitter. Lines marked with cs represent one-way delay non-AP to AP and sc represent the opposite direction.

both data and ACK packets AIFS and estimation of back-off time. See the following equations for details.

$$t_{\text{frame}}(\text{bytes}) = t_{\text{plcp}} + 8 \cdot \text{bytes}/\text{bitrate} \quad (1)$$

$$b(\text{payload}) = \text{payload} + l_{\text{udp,ip,llc,mac,fcs}} \quad (2)$$

$$t_{\text{tx}}(p) = t_{\text{bkoff}} + t_{\text{frame}}(b(p)) + t_{\text{sifs}} + t_{\text{frame}}(\text{ack}) \quad (3)$$

$$t_B = \lfloor B/\text{MTU} \rfloor t_{\text{tx}}(\text{MTU}) + t_{\text{tx}}(B \bmod \text{MTU}) \quad (4)$$

The estimation of backoff time is based on AC parameters and is multiplied by an empirical constant C_i different for each AC which should roughly represent the influence of collisions: $t_{\text{bkoff},i} = C_i \cdot t_{\text{slot}} \cdot (AIFS_N[i] + CW_{\text{min}}[i]/2)$. Because of this simple estimation, this test is not very precise in general, as the relationships between the traffic and number of collisions are more complex and retransmission time is not counted.

For each of already accepted and new contracts, we calculate the length of bus occupancy t_{B_k} and divide it by the contract period T_k . Then, these numbers (partial utilization values) are summed to form the total utilization

$$U = \sum_{\forall k} \frac{t_{B_k}}{T_k} \quad (5)$$

If the total utilization U is less than some value (e.g. 80%) the new contract is accepted, otherwise it is rejected.

To illustrate properties of this test we have performed three experiments to measure the behavior of WLAN communication and the results were compared with the results given by the utilization based test. After tuning the empirical constants, we have found quite good match (which doesn't mean the match will be of the same quality for different experiments).

In the experiments, we have measured the saturation bandwidth of AC_BE as a function of the size of packets in AC_VO, AC_VI and AC_BE respectively (see Fig. 9). In each of the three tests, the following streams were generated: AC_VO – 100 kbps, AC_VI – 100 kbps and AC_BE

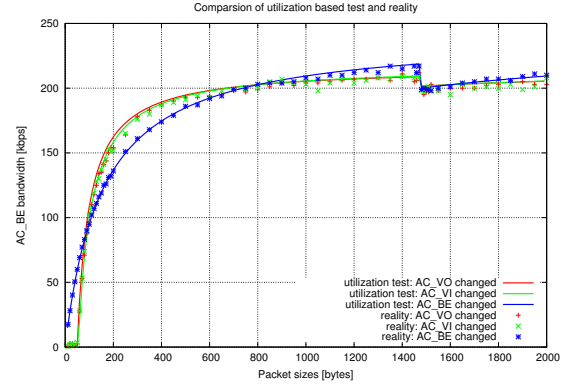


Figure 9. Comparison of the utilization based test and measured results for three different experiments.

– 500 kbps (saturation). Packet size in one stream was being changed and the other streams was formed by packets of 800 bytes (UDP data). The real (saturation) bandwidth of AC_BE was measured.

Figure 9 shows the measured results together with the results generated by the utilization test, where we have used binary chopping algorithm to find the maximum bandwidth of AC_BE stream. The solid lines correspond to AC_BE bandwidth under theoretical utilization of 95% and the points represent the measured bandwidth for a particular experiment. The step at packet size of 1472 is caused by IP protocol fragmentation (at MTU size).

5. Conclusions and future work

This paper presents initial work undertaken in order to have wireless LANs supported by a contract framework. Our future work will head towards having a better testbed with more stations. Further, we will improve the admission test to provide reasonable accuracy for all possible communication scenarios. Also, we would like to implement another admission test based on [5] and compare it with our simple test. Finally support for changing bit-rates must be added.

References

- [1] K. Tindell and J. Clark, “Holistic schedulability analysis for distributed hard real-time systems”, *Microprocess. Microprogram.*, vol. 40, no. 2-3, pp. 117–134, 1994.
- [2] M. González Harbour and M. Tellería de Esteban, “Architecture and contract model for integrated resources II”, Deliverable of the FRESCOR project (D-AC2v2), 2008.
- [3] IEEE 802.11 — WG Reference number ISO/IEC 8802-11:1999(E), “IEEE Std 802.11e”, 2005.
- [4] F. Guo, *Implementation Techniques for Scalable, Secure and QoS-Guaranteed Enterprise-Grade Wireless LANs*, PhD thesis, Stony Brook University, August 2006.
- [5] P. Engelstad and O. Østerbø, “The delay distribution of IEEE 802.11e EDCA and 802.11 DCF”, in *Performance, Computing, and Communications Conference*, 2006.