METHOD TO EMULATE THE L-BAND DIGITAL AERONAUTICAL COMMUNICATION SYSTEM FOR SESAR EVALUATION AND VERIFICATION

T. Gräupl, German Aerospace Center (DLR), Institute of Communications and Navigation, Oberpfaffenhofen-Wessling, Germany

M. Mayr, University of Salzburg, Department of Computer Science, Salzburg, Austria

Abstract

The VHF voice communication system currently used for air traffic control is experiencing increasing capacity problems. The "L-band Digital Aeronautical Communications System" is an upcoming technology providing an aeronautical datalink outside of the VHF band. The objective of this paper is to develop a method to emulate its communication performance. We developed a formal model of the system and implemented it on the basis of the *dummynet* network emulation tools. This implementation was deployed in a networking appliance and measured in a test-bed network. The results indicate that the emulator provides the performance predicted by simulations and is suitable to evaluate and verify protocols and applications envisioned to utilize this datalink.

Introduction

Today's VHF voice-based air-ground communication system for tactical aircraft guidance is suffering from the VHF band's increasing saturation in high density areas [1]. It is supplemented by aeronautical datalinks also operating in the VHF band, most notably the Aircraft Communications Addressing and Reporting System (ACARS) and VHF Digital Link Mode 2. However, ACARS is already heavily used for the business communication of the airlines, and VHF Digital Link Mode 2 is scarcely deployed and suffers from operational problems [2]. In addition, its further deployment is hindered by the heavy use of the VHF band.

The L-band Digital Aeronautical Communications System (LDACS) is a broadband air-ground datalink proposed to supplement the VHF communication infrastructure in the L-band [3] [4]. It is designed to provide air-ground data communication with optional support for digital voice. It is a cellular broadband system based on Orthogonal Frequency-Division Multiplexing (OFDM) technology [5] and supports quality-of-service taking the requirements of aeronautical services into account [6]. It shares many technical features with 3G and 4G wireless communications systems.

Computer simulations have assessed the expected performance of LDACS. Multiple independent simulation campaigns were conducted by Gräupl et al. [7] [8] [9], Micallef et al. [10], and Ayaz et al. [11] predicting similar performance figures. Prototype implementations in hardware are under way [12] [13], but focus on the physical layer.

The current unavailability of full-system prototypes leaves a gap preventing the evaluation of the overall system performance in realistic test-bed networks. Full-system prototypes will become available in the next years. However, the performance of the air traffic management applications and protocols developed in the Single European Sky Air Traffic Management Research Programme (SESAR) shall be evaluated and verified as soon as possible.

The objective of this paper is to develop a method to emulate the LDACS datalink, such that it matches the LDACS user-plane performance in terms of bandwidth, latency, and loss, and can be used in the test-bed networks set up by SESAR project 15.2.4 evaluating and verifying the multilink concept, quality of service management, and network mobility concepts developed within that project.

Definitions

In this paper, we follow Carson et al. [14] in the definition of network emulation: Emulation is a semi-synthetic environment for testing real networking code (i.e. applications, protocols). It is semi-synthetic in the sense that a real network implementation is supplemented with means for introducing synthetic delays and faults.

Method

The scope of the LDACS system emulator is to emulate the perceived user-plane performance (latency, bandwidth and loss) of the wireless LDACS data-link at the network layer. Multiple aircraft and multiple wireless connections of varying signal quality may be emulated. The internal management and mobility protocols of the LDACS access network are not emulated.

The emulation is accomplished by the implementation of a formal model of the LDACS protocols in network emulation software tools. This network emulation software is installed on a personal computer to create an LDACS system emulator appliance to be embedded into the test-bed networks. It provides Ethernet interfaces towards the on-board network and the ground network. Several on-board networks may be connected to the "airborne" Ethernet interface to emulate multiple aircraft. Only one ground network may be connected to the "ground" interface to emulate the ground telecommunications infrastructure. The LDACS datalink emulation takes place between the "airborne" and the "ground" Ethernet interfaces.

Model

LDACS will be one of several wireless access networks connecting aircraft to the aeronautical telecommunications network displayed in Figure 1. The LDACS sub-network contains several ground stations, each of them providing one LDACS radio cell.

In order for the LDCAS system emulator to emulate the performance of the LDACS sub-network correctly, it has to take the LDACS radio protocol stack into account. The LDACS air-ground data-link sub-system is a cellular system with a star-topology connecting aircraft to ground-stations with a full duplex radio link. Each ground-station is the centralized instance controlling all air-ground communications within its radio cell.

The LDACS protocol stack defines two layers, the physical layer and the data link layer. The LDACS system emulator emulates the physical layer in terms of statistical frame error rates representing the perceived link quality. A formal model of the medium access and logical link control protocols emulates the data-link layer.

The physical layer provides the means to transfer data over the radio channel. The LDACS ground-station supports bi-directional links to multiple aircraft under its control. The forward link direction (FL; ground-to-air) and the reverse link direction (RL; air-to-ground) are separated by frequency division duplex. Forward link and reverse link use a 500 kHz channel each. The ground-station transmits a continuous stream of OFDM symbols on the forward link. In the reverse link different aircraft are separated in time and frequency using a combination of orthogonal frequency-division multiple-access and time-division multiple-access. Aircraft thus transmit discontinuously on the reverse link with radio bursts sent in precisely defined transmission opportunities allocated by the ground-station.

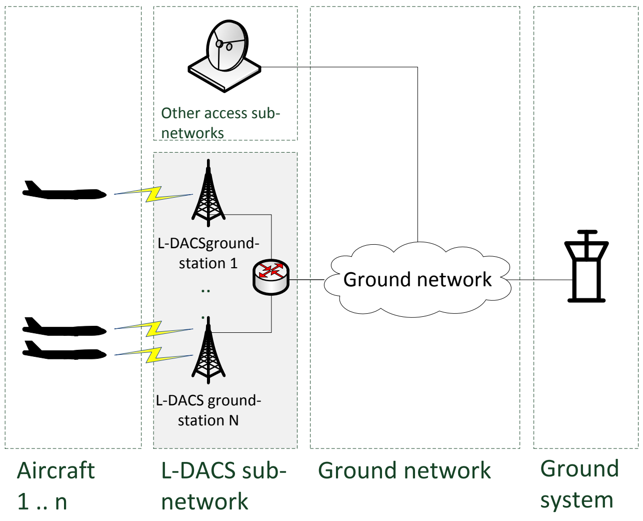

Figure 1: LDACS sub-network (highlighted) within the aeronautical telecommunications network.

LDACS structures the OFDM symbols of the physical layer into multi-frames (MF) of 58.32 ms duration. Note that the average multi-frame length is 60 ms if the random access slot appearing every 240 ms is taken into account. The random access slot is used for the initial log-in of aircraft and not relevant within the context of this paper.

Each forward link multi-frame comprises nine OFDM frames. Frames are sub-sets of 814 OFDM symbols within the forward link multi-frame that are encoded together (cf. Figure 2, lower channel). A variable number of frames at the start of each forward link multi-frame are designated as Common Control (CC) slot. The remaining frames create the data slot. The net bandwidth in the forward link data slot is

291.2 kbit/s per cell in the basic physical layer configuration (QPSK, coding rate 0.5) and assuming the default common control slot size of one frame.

The reverse link multi-frame is divided into sub-sets of 134 OFDM symbols called tiles (cf. Figure 2, upper channel). Each reverse link multi-frame comprises a Dedicated Control (DC) slot and a data slot. The reverse link dedicated control slot starts with synchronization symbols and the first two tiles. Its length is variable between 2 tiles and 52 tiles. The remaining tiles create the reverse link data slot. The coding and modulation of tiles in the reverse link data slot can either be fixed for the entire LDACS cell or be changed dynamically by the ground-station. The LDACS system emulator assumes fixed coding and modulation in the basic physical layer configuration (QPSK, coding rate 0.5). Taking the default dedicated control slot size into account (52 tiles), the net user data rate in the reverse link data slot is 220 kbit/s.

Forward link and reverse link multi-frames are offsetted by 30 milliseconds to interleave the control channel slots in time as illustrated in Figure 2.

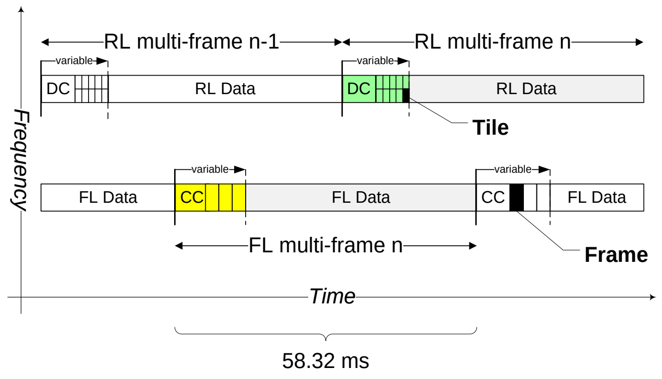

Figure 2: LDACS multi-frame structure.

The data-link layer provides the necessary protocols to facilitate concurrent and reliable data transfer for multiple users. The LDACS data link layer is organized in two sub-layers: The medium access sublayer and the logical link control sub-layer. The medium access sub-layer manages the organization of transmission opportunities in slots of time and frequency (i.e. frames and tiles). The logical link control sub-layer provides reliable and acknowledged point-to-point logical channels between the aircraft and the ground-station using an automatic repeat request protocol.

Data packets are scheduled for transmission in the common control slot at the beginning of each multi-frame. The common control slot contains the mapping of the data packets to the frames and tiles of the forward link and reverse link data slots. Logical link control sub-layer packets transmitted in the data slots are decoded at the end of the data slots. This is illustrated in Figure 3.

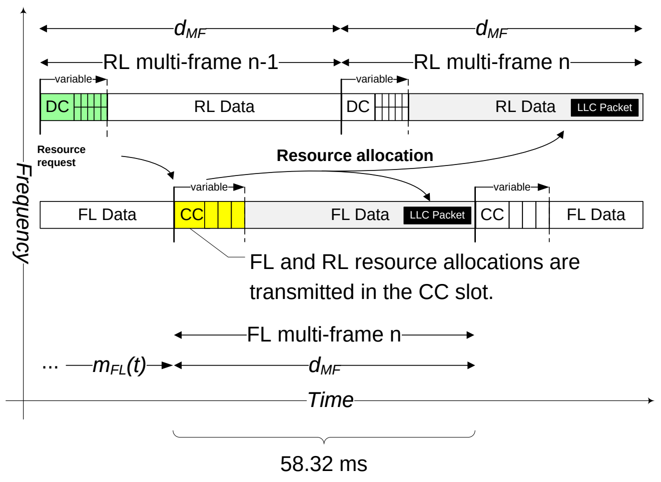

Figure 3: LDACS medium access.

Forward link transmissions are scheduled by the ground-station according to its local transmission queue. If no retransmission is triggered by the logical link control protocol, the forward link one-way latency l_{FL} of a packet sent at time t_0 is

$$l_{FL}(t_0) = m_{FL}(t_0) + d_{MF}$$

where $m_{FL}(t_0)$ is the time until the start of the next common control slot and d_{MF} is the length of the multi-frame.

If a retransmission is performed by the logical link control protocol, the one-way latency is increased by the forward link retransmission latency r_{FL}. Forward link transmissions have to be acknowledged by the aircraft in the dedicated control slot, which is $d_{CC\rightarrow DC}$ after the end of the data slot. If no acknowledgement is received, the packet is scheduled for retransmission at the start of the next common control slot $d_{DC\rightarrow CC}$ later. However, aircraft are polled in the common control slot to send acknowledgements in the dedicated control slot in round-robin. The reverse link medium access cycle is the time required by the ground-station to poll all aircraft in the cell. Poll commands are sent in the common control slot. Aircraft reply with an acknowledgement or resource request in a tile of the dedicated control slot indicated in the poll command.

The time until the next dedicated control slot for which the aircraft is polled is denoted $m_{RL}(t)$. The actual retransmission requires the duration of another multi-frame. The forward link retransmission latency is then

$$r_{FL}(t_1) = m_{RL}(t_1) + d_{DC\rightarrow CC} + d_{MF}$$

where t_1 is the time of the first possible occurrence of the dedicated control slot

$$t_1 = t_0 + l_{FL}(t_0) + d_{CC\rightarrow DC}.$$

Since

$$d_{CC\rightarrow DC} + d_{DC\rightarrow CC} = d_{MF}$$

and $m_{RL}(t)$is always an integer multiple of d_{MF} if t is the start of the dedicated control slot (as in the case of t_1), the total forward link one-way latency is

$$L_{FL}(t) = m_{FL}(t) + (1 + \delta_{RX}(1 + n)) \times d_{MF}$$

with $\delta_{RX} \in \{0,1\}$ indicating a retransmission, and n according to the length of the reverse link medium access cycle.

The reverse link uses a bandwidth on demand scheme. Aircraft have to request channel resources from the ground-station before they can transmit data packets in the reverse link data slot. To this purpose aircraft are polled by the ground-station in the common control slot for their resource request in round-robin. In the default configuration 52 aircraft can be polled to respond in the dedicated control slot. The resource request sent in the indicated tile is an aggregate request for all resources needed by the aircraft. Having received the resource request of all polled aircraft, the ground-station assigns resources using an appropriate scheduling algorithm. There are two published implementations of LDACS resource schedulers by Gräupl [15] and Ayaz et al. [16]. Both implementations are round-robin schedulers.

Note that polling for resource requests may take more than one multi-frame if the number of users in the cell exceeds 52. Multi-frame n-1 and multi-frame n in Figure 3 would then be delayed by the according number of multi-frames required to poll the aircraft in round-robin. The time until the next dedicated control slot for which the aircraft is polled is denoted $m_{RL}(t)$.

Aircraft request resources in the dedicated control slot at the beginning of their medium access cycle. Resources in the next RL data slot are allocated by the ground-station in the common control slot. Packets transmitted in the data slot are decoded at the end of the data slot which is also the end of the multi-frame (cf. Figure 3). If no retransmission is triggered by the logical link control protocol, the reverse link one-way latency l_{RL} of a packet sent at time t_0 is then

$$l_{RL}(t_0) = m_{RL}(t_0) + d_{DC\rightarrow CC} + d_{CC\rightarrow DC} + d_{MF}$$

where $m_{RL}(t_0)$ is the time until the start of the next dedicated control slot, $d_{DC\rightarrow CC}$ is the time from the start of the dedicated control slot to the common control slot, $d_{CC\rightarrow DC}$ is the time from the start of the common control slot to the dedicated control slot, and d_{MF} is the length of the multi-frame.

If a retransmission is performed by the logical link control protocol, the one-way latency is increased by the reverse link retransmission latency r_{RL}. Reverse link transmissions have to be acknowledged by the ground-station in the common control slot, which is $d_{DC\rightarrow CC}$ after the end of the data slot. If no acknowledgement is received, the aircraft will request the resources for a retransmission in the next dedicated control slot for which it is polled. The retransmission requires the duration of the resource allocation in the common control slot and the transmission of the packet in the following reverse link data slot. The forward link retransmission latency is then

$$r_{RL}(t_1) = m_{RL}(t_1) + d_{DC\rightarrow CC} + d_{CC\rightarrow DC} + d_{MF}$$

where t_1 is the time of the start of the dedicated control slot after the missing acknowledgement

$$t_1 = l_{RL}(t_0) + d_{DC\rightarrow CC} + D_{CC\rightarrow DC}.$$

Since

$$d_{CC\rightarrow DC} + d_{DC\rightarrow CC} = d_{MF}$$

and $m_{RL}(t)$is always an integer multiple of d_{MF} if t is the start of the dedicated control slot (as in the case of t_1), the total reverse link one-way latency is

$$L_{RL}(t) = m_{RL}(t) + (2 + \delta_{RX}(n + 3)) \times d_{MF}$$

with $\delta_{RX} \in \{0,1\}$ indicating a retransmission, and n according to the length of the reverse link medium access cycle.

Tools

The formal model of the LDACS protocol is implemented in a tool-chain of network emulation software. The LDACS sub-system emulator uses a combination of established network emulation tools and custom tools developed for the LDACS system emulator.

The basis of the LDACS system emulator is the *FreeBSD* version 10 operating system and its tools. The Unix-like operating system *FreeBSD* is a direct descendant of Berkeley Unix and is the most widely used BSD derivate. The LDACS protocol model is implemented using the stateful firewall *ipfw* and the *dummynet* traffic shaper included with the operating system.

ipfw is the stateful firewall of *FreeBSD*. It supports IPv6 and is included in the standard *FreeBSD* installation as a kernel module. *dummynet* is the traffic shaper module of the *ipfw* firewall. Its three main tools for network emulation are pipes, queues and schedulers. The protocol model is implemented with these three tools.

Pipes emulate links with a single queue and a first-in first-out scheduler. They can be configured to limit the bandwidth and add delays. Additionally, a pipe can be configured with a probability value or a probability distribution, activating a random decision process for each packet e.g. to add additional delay with a certain probability.

A queue is a buffer used to feed a scheduler. Queue sizes can be configured either as number of stored packets or with the queue size in bytes. Each queue needs to be linked to a pipe. Several queues can be linked to the same pipe using scheduler and flow masks to create dynamic queues. If no scheduler is configured for a pipe, the standard scheduler is used.

The standard *dummynet* scheduler uses the WFQ2+ worst-case weighted fair queuing algorithm, with $O(\log n)$ processing cost per packet. In accordance with the published LDACS protocol implementations a deficit round robin scheduler was used, that is also included with *dummynet*. This scheduler has a processing cost of $O(1)$.

In addition to the network emulation tools the LDACS system emulator uses the *ping6* and *tg* tools to create data traffic loads for the emulator. *ping6* is a utility which uses the ICMP protocol's ECHO_REQUEST IPv6 datagram message to trigger an ICMP ECHO_RESPONSE from a target host. The requested quality of service, packet size and interval between sent packets can be configured. It is used as a simple delay and bandwidth-testing tool.

tg is a custom Java program for generating background traffic. It is used to emulate the traffic of large numbers of aircraft. The message size, traffic volume, requested quality of service, and the probability distribution of the classes of service is configurable.

The implementation of the LDACS system emulator is further supported by the *bash* and *ssh* utilities for (remote) command scripting. The Bourne-again shell *bash* is a command-line interpreter and command language used by many Unix-like operating systems. It enables usage of scripted command execution. *ssh*, the OpenSSH secure shell client is used to issue remote commands to the LDACS system emulator.

Implementation

To emulate the LDACS sub-network performance, the model of the LDACS protocol stack is mapped onto *dummynet* queues, pipes, and schedulers assisted by *bash* scripts.

The LDACS system emulator has two emulation paths: Full emulation and partial emulation. This is done to facilitate scenarios with large aircraft populations without having to actually deploy all aircraft in the testbed network. Instead, it is sufficient to deploy only the fully emulated aircraft used for measurements e.g. as appliances connected to the testbed network. The remaining aircraft population is emulated by the background traffic generator *tg* in software to emulate their influence on the overall communication performance. Only the first path provides the full emulation of the LDACS protocol model.

The full emulation path involves four stages: The first stage adds retransmission delays induced by the logical link control protocol with given probabilities. This stage is skipped by the background traffic, because of the processing cost of the random packet processing. The second stage models the timing of the multi-frames and the medium access cycle. The third stage models the resource scheduling of the ground-station. The last stage applies the bandwidth limitations of the wireless channel and adds any constant delays. The tool-chain is illustrated in Figure 4. The order of the stages is as required by *dummynet*.

The emulation tool-chain is unidirectional. For full-duplex communication it has to be instantiated twice for each emulated cell: Once for the forward link, and a second time for the reverse link. Ground-to-air packets are always passed to the LDACS system emulator on the “ground” Ethernet interface connected to the ground-network. Air-to-ground packets are always passed to the emulator on the “airborne” Ethernet interface connected to the on-board networks. Additionally the IPv6 addresses of the aircraft and the ground station are taken into account. It is therefore always possible for *ipfw* to route packets according to the interface and the source address to the correct *dummynet* forward link tool-chain or reverse link tool-chain.

When several LDACS radio cells are emulated, *ipfw* is used to route packets according to the IPv6 address of the aircraft to the correct pair of *dummynet* tool-chains implementing the cell. The assignment of aircraft to radio cells can be changed at run time by issuing scripted *ipfw* commands via *ssh*.

Retransmission

The implementation of the retransmission model uses pipes configured to add retransmission delays with a probability given by the link quality. There is one pipe for each class of service. Three classes of service are configured in the LDACS system emulator: High, medium, and low priority. The probability distribution of the delay added by these pipes is calculated by assuming that retransmissions happen as stochastically independent events. This leads to the following probability distribution:

$$P(\{k \text{ retransmissions}\}) = (1-p) * p^k$$

where p is the packet error probability and k the number of retransmissions.

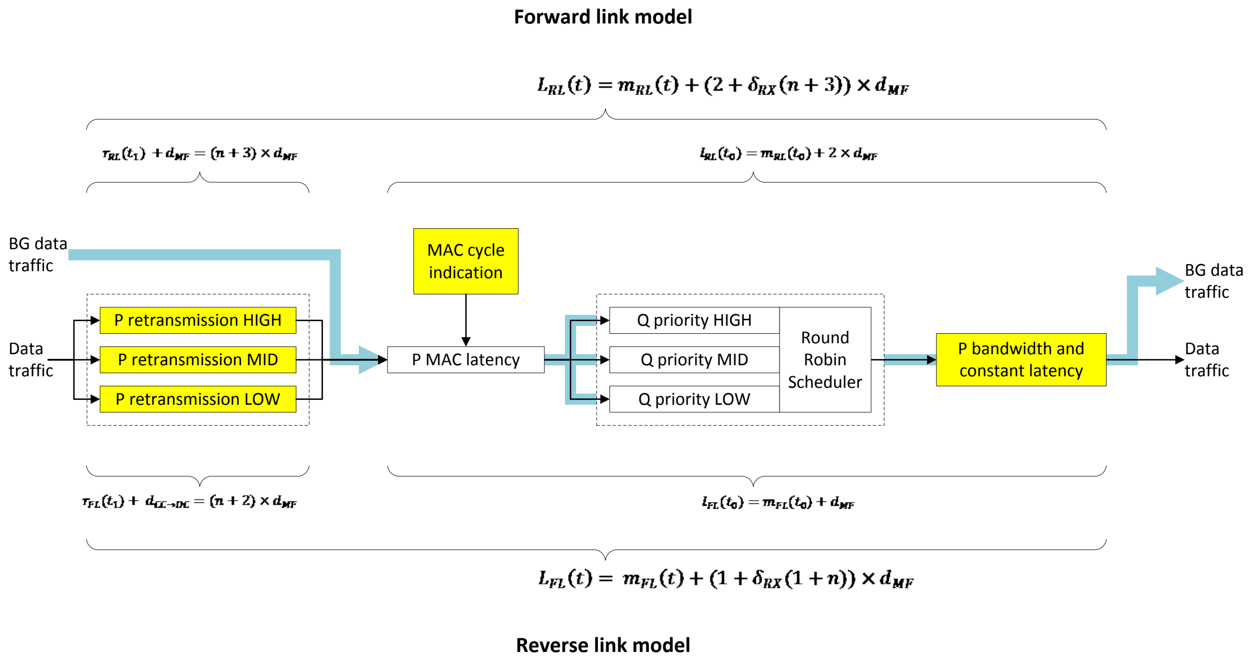
Forward link model
$L_{RL}(t) = m_{RL}(t) + (2 + \delta_{RX}(n+3)) \times d_{MF}$
$\tau_{RL}(t_1) + d_{MF} = (n+3) \times d_{MF}$
$l_{RL}(t_0) = m_{RL}(t_0) + 2 \times d_{MF}$
BG data traffic
Data traffic
P retransmission HIGH
P retransmission MID
P retransmission LOW
MAC cycle indication
P MAC latency
Q priority HIGH
Q priority MID
Q priority LOW
Round Robin Scheduler
P bandwidth and constant latency
BG data traffic
Data traffic
$\tau_{FL}(t_1) + d_{CC \to DC} = (n+2) \times d_{MF}$
$l_{FL}(t_0) = m_{FL}(t_0) + d_{MF}$
$L_{FL}(t) = m_{FL}(t) + (1 + \delta_{RX}(1+n)) \times d_{MF}$
Reverse link model

Figure 4: Unidirectional implementation of the LDACS protocol model. Retransmission model is on the left, transmission model on the right. For bidirectional emulation forward link and reverse link instances are created with different parameters. For each additional cell an additional pair of instances is required. Configurable elements are highlighted. Background (BG) traffic is indicated with bold arrows.

Transmission

The LDACS medium access behavior is emulated using a *dummynet* pipe according to Armitage et al.'s approach [17]: The configuration of the pipe is regularly updated by an external *bash* script and the *ipfw* command. In order to create delays synchronized to the LDACS frame structure and medium access cycle, the pipe is set to the duration of the medium access cycle for a short time and then set to zero delay for the rest of the medium access cycle.

Packets received during the short maximum delay phase delay following packets for the length of the medium access cycle. A 120 ms medium access cycle (equivalent to 104 aircraft cell population) is, for example, realized by setting the pipe's delay to 120 ms for a two millisecond duration and then setting the pipes delay to 0 ms for the remaining 118ms of the medium access cycle. This is repeated at the start of each medium access cycle. To ascertain that data traffic sent to the emulator is always delayed as intended, the LDACS system emulator injects dummy packets of negligible size as "blocking packets" into the pipe during the maximum delay phase. This is accomplished with *ping6*.

Table 1: Parameters of the forward link model.

	Retransmission **(stage 1; applied with probability *p*)**	**Transmission** **(stage 2-4)**
Model	$r_{FL}(t_1) + d_{CC\rightarrow DC} = (n+2) \times d_{MF}$	$l_{FL}(t_0) = m_{FL}(t_0) + d_{MF}$
d_{MF}	60 ms	60 ms
$m_{FL}(t)$		Time until start of next FL multi-frame. A FL multi-frame starts every d_{MF}.
n	Maximum number of multi-frames after the transmission until the next DC slot is scheduled for the A/C in the MAC-cycle: 50 A/C: n=0 100 A/C: n=1 150 A/C: n=2	
p	For BER 10^{-7}, 10^{-5}, 5×10^{-5} p is 0.01%, 1%, 5% for 135 Bytes packet size.	

We follow Ayaz et al. [16] by implementing the resource scheduling in the ground-station with a deficit round-robin scheduler. We use the implementation provided by *dummynet*; the high, medium, and low classes of services are weighted 100:10:1.

The last stage applies the bandwidth limitations of the wireless channel and adds any constant delays. The forward link bandwidth is 291.2 kbit/s per cell with the default common control slot configuration and the basic physical layer configuration. This bandwidth is reduced by 5% to account for the retransmission overhead caused by the maximum packet error rate, and by further 10% to account for the maximum logical link control protocol overhead. This results in a worst case net forward link bandwidth of 247.52 kbit/s. The reverse link bandwidth is 220 kbit/s per cell cell with the default dedicated control slot configuration and the basic physical layer configuration resulting in a worst case net reverse link bandwidth of 187 kbit/s according to the same calculation as above.

Table 2: Parameters of the reverse link model.

	Retransmission **(stage 1; applied with probability *p*)**	**Transmission** **(stage 2-4)**
Model	$r_{RL}(t_1) + d_{MF} = (n+3) \times d_{MF}$	$l_{RL}(t_0) = m_{RL}(t_0) + 2 \times d_{MF}$
d_{MF}	60 ms	60 ms
$m_{RL}(t)$		Time until start of next MAC-cycle. The MAC-cycle length is 50 A/C: 60 ms 100 A/C: 120 ms 150 A/C: 180 ms
n	Number of multi-frames after the transmission until the next DC slot is scheduled for the A/C in the MAC-cycle: 50 A/C: n=0 100 A/C: n=0 150 A/C: n=1	
p	For BER 10^{-7}, 10^{-5}, 5×10^{-5} p is 0.01%, 1%, 5% for 135 Bytes packet size.	

The processing delay of *dummynet* is taken into account by empirically reducing the constant delays by few milliseconds. This is hardware dependent and not necessary if small latency errors are of no concern.

The parameters of the formal model are listed in Table 1 and Table 2 for the scenarios to be evaluated in SESAR project 15.2.4.

Results

The objective of this paper was to develop a method to emulate the LDACS user-plane performance in terms of bandwidth, latency, and loss.

The net bandwidth of the LDACS system emulator was measured with *iperf*. We measured 237 kbit/s on the forward link and 179 kbit/s on the reverse link in a cell with no background aircraft population and 0.01% packet error rate.

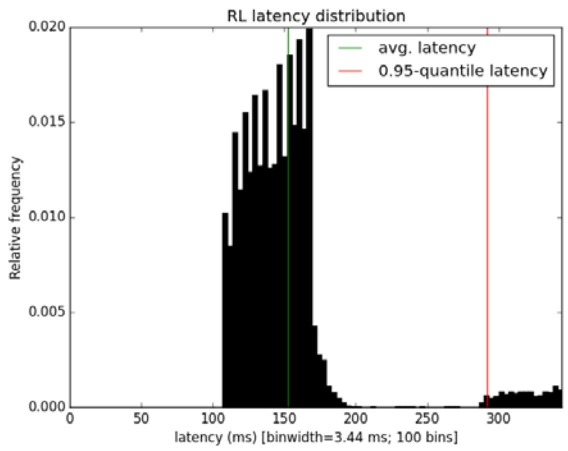

Figure 5: Histogram of the reverse link latency distribution from the emulation; 50 aircraft per cell; 5% packet error rate.

We measured the latency of the LDACS system emulator in a representative scenario defined for SESAR project 15.2.4. The scenario comprises one radio cell and one fully emulated aircraft. Only the fully emulated aircraft was measured. The background aircraft population of the cell was 50 aircraft. Data traffic was produced according to Ehammer et al.'s [18] characterization of aeronautical data traffic patterns. Each aircraft produced 1 kbit/s of data traffic on the forward link and 1 kbit/s of data traffic on the reverse link. The packet size was set to 135 Bytes (including the IPv6 header). Packets were generated with exponentially distributed inter-arrival times. The distribution of the requested classes of service was 10% high priority, 30% medium priority, and 60% low priority. The wireless link quality was set to bit error rates of 10^{-7}, 10^{-5}, and 5×10^{-5} which is equivalent to packet error rates of 0.01%, 1%, and 5% for the given packet size.

Figure 5 shows the distribution of the reverse link latency in this scenario for 5% packet error rate. We measured the one way latency by one-way emulation. The Ethernet delay in the other direction is neglected. The result was corrected for a *dummynet* processing delay of 13 ms on the forward link and 16 ms on the reverse link. No packets were lost in the measurement.

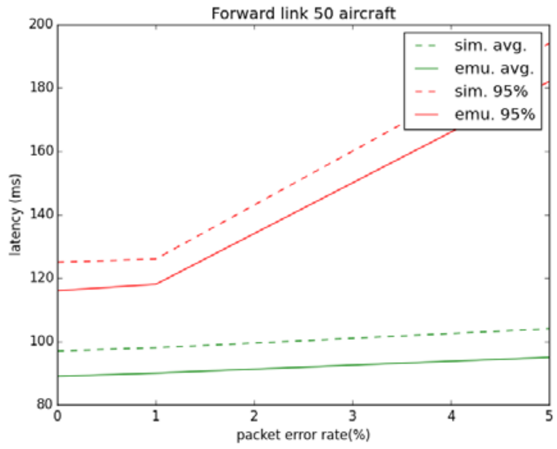

Figure 6: Forward link latency for 50 aircraft cell population.

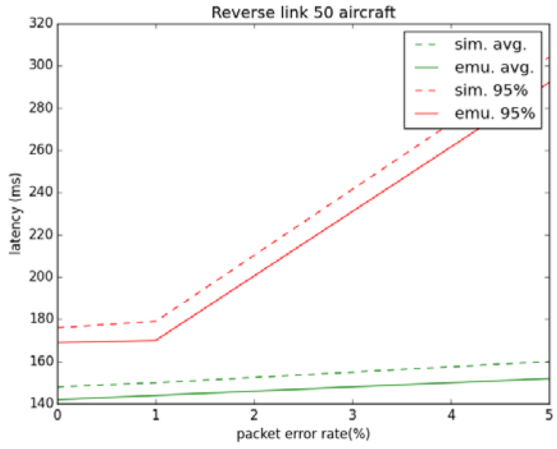

Figure 7: Reverse link latency for 50 aircraft cell population.

Figure 6 and Figure 7 show the measurement results of the average latency and the 95% percentile of the latency for packet error rates of 0.01%, 1%, and 5%.

Discussion

The results show clearly that the LDACS system emulator matches the predicted LDACS user-plane performance in the measured scenarios in terms of bandwidth, latency and loss.

The maximum user-plane bandwidth provided by the LDACS system emulator is slightly lower than expected. However, the *iperf* tool does not take TCP overhead into account.

The packet loss characteristic is as expected. As LDACS uses a reliable logical link control protocol no packet losses should occur. The measurements matched this expectation.

The measured average and 95% percentile of the one-way latency matched the predicted results very well for different bit error rates (cf. Figure 6 and Figure 7). We used our simulator from [8] for the prediction.

The distribution of the one-way latency values resembles the distribution predicted by the simulation (cf. Figure 5). For comparison the reverse link latency distribution of the simulation of the same scenario is shown in Figure 8. The two main peaks of the distribution are both present. Packets successfully transmitted in the first attempt cause the left peak. Retransmitted packets cause the peak on the right. The small steps to the left and right of the first peak are only present in the simulation results. This is caused by differences in the ground-station scheduler implementation. The scheduler of the simulation allows high priority packets to "steal" bandwidth allocated to lower priority packets, spreading the variance of the distribution. The skew of the left peak that can only be found in the emulation measurements is, in our opinion, due to the bandwidth limitation queue in stage 4 of the emulation that is not present in the simulation.

In general, we argue that the results indicate that our method is suitable to emulate the LDACS datalink performance in sufficient accuracy for the evaluation and verification of the envisioned air traffic management protocols and applications.

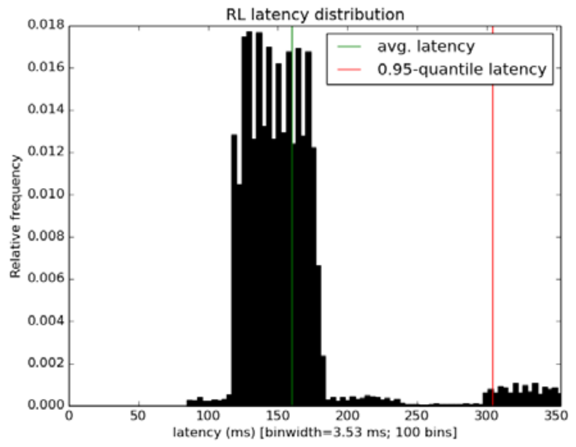

Figure 8: Histogram of the reverse link latency distribution from the simulation; 50 aircraft per cell; 5% packet error rate.

It was unexpected for us that these results could only be achieved with the *FreeBSD* operating system. Early experiments with Linux network emulation software showed considerable performance problems with complex firewall configurations as well as missing functionality for network emulation.

The method developed in this paper is limited by the unavailability of full LDACS implementations. The results can only be assessed relative to results predicted by simulations. The simulations may, however, not capture all aspects of an hardware LDACS implementation.

In addition, we observed limitations in the *dummynet* schedulers. While all queues have at least one packet waiting, the schedulers work as expected. However, the scheduler seems not to consider information on already scheduled packets and uses only the current queue state for its scheduling decisions. This induces unwanted behavior when the bandwidth limitation of the pipe is not reached and the queues remain empty. This is partially compensated by the external manipulation of the medium access delay pipe, whose delay is periodically set to high values filling the queue.

Our method and its implementation in a networking appliance provides SESAR project 15.2.4 with the required means to evaluate and verify the multilink concept, quality of service management, and network mobility concepts within a test-bed network.

Conclusion

The objective of this paper was to develop a method to emulate the LDACS user-plane performance in terms of bandwidth, latency, and loss.

We developed a formal model of LDACS and implemented our method in an appliance running the *FreeBSD* operating system. Measurements from this implementation were compared with predicted results from LDACS computer simulations.

The results indicate clearly that our method is suitable to emulate the LDACS datalink performance with sufficient accuracy for the evaluation and verification of the air traffic management protocols and applications envisioned by SESAR.

Acronyms and Abbreviations

ACARS	Aircraft Communications Addressing and Reporting System
BER	Bit Error Rate
CC	Common Control
DC	Dedicated Control
FL	Forward Link
ICMP	Internet Control Message Protocol
IPv6	Internet Protocol version 6
LDACS	L-band Digital Aeronautical Communication System
MAC	Medium Access
MF	Multi-Frame
OFDM	Orthogonal Frequency-Division Multiplexing
QPSK	Quadrature Phase-Shift Keying
RL	Reverse Link
SESAR	Single European Sky Air Traffic Management Research Programme

Acknowledgements

Parts of this work were performed while Thomas Gräupl was an employee of the University of Salzburg.

References

[1] B. Kamali, “An overview of VHF civil radio network and the resolution of spectrum depletion,” in *2010 Integrated Communications, Navigation, and Surveillance Conference Proceedings, ICNS 2010*, 2010.

[2] C. Gandolfi, E. Mora-Castro, J. Myllarniemi, and F. Copigneaux, “Technical issues in the implementation of Regulation (EC) No 29/2009 (Data Link),” 2014.

[3] M. Schnell, U. Epple, D. Shutin, and N. Schneckenburger, “LDACS: future aeronautical communications for air-traffic management,” *Commun. Mag. IEEE*, vol. 52, no. 5, pp. 104–110, 2014.

[4] M. Sajatovic, B. Haindl, M. Ehammer, T. Gräupl, M. Schnell, U. Epple, and S. Brandes, “L-DACS1 System Definition Proposal,” *Eurocontrol Study, Ed.*, vol. 1, 2009.

[5] S. Brandes, U. Epple, S. Gligorevic, M. Schnell, B. Haindl, and M. Sajatovic, “Physical layer specification of the L-band Digital Aeronautical Communications System (L-DACS1),” in *Proceedings of the 2009 Integrated Communications, Navigation and Surveillance Conference, ICNS 2009*, 2009.

[6] T. Graupl, M. Ehammer, and C.-H. Rokitansky, “L-DACS 1 data link layer design and performance,” *2009 Integr. Commun. Navig. Surveill. Conf.*, 2009.

[7] T. Gräupl, M. Ehammer, and C. H. Rokitansky, “Link-layer quality of service in the L-band digital aeronautical communication system B-amc,” in *AIAA/IEEE Digital Avionics Systems Conference - Proceedings*, 2008.

[8] T. Gräupl and M. Ehammer, “LDACS1 data link layer evolution for ATN/IPS,” in *AIAA/IEEE Digital Avionics Systems Conference - Proceedings*, 2011.

[9] T. Gräupl and M. Ehammer, *The LDACS1 Link Layer Design*. InTech, 2011.

[10] J. Micallef, A. Burrage, and A. Parkinson, "Independent Validation of L-DACS/1 Performance through Simulation Modelling," 2009.

[11] S. Ayaz, F. Hoffmann, U. Epple, R. German, and F. Dressler, "Performance evaluation of network mobility handover over future aeronautical data link," *Comput. Commun.*, vol. 35, no. 3, pp. 334–343, 2012.

[12] N. Franzen, A. Arkhipov, and M. Schnell, "L-DACS1 physical layer laboratory demonstrator," in *2010 Integrated Communications, Navigation, and Surveillance Conference Proceedings, ICNS 2010*, 2010.

[13] N. Fistas and M. Schnell, "ICAO AERONAUTICAL COMMUNICATIONS PANEL (ACP) FOURTH MEETING OF THE WORKING GROUP OF THE WHOLE: LDACS1 Physical Layer Laboratory Demonstrator," 2011.

[14] M. Carson and D. Santay, "NIST Net: a Linux-based network emulation tool," *ACM SIGCOMM Comput. Commun. …*, vol. 33, no. 3, pp. 111–126, 2003.

[15] T. Gräupl, "Analysis, Design and Evaluation of the User Plane Quality-of-Service Protocols in Future Terrestrial Digital Aeronautical Communication Systems," University of Salzburg, 2011.

[16] S. Ayaz, F. Hoffmann, R. German, and F. Dressler, "Analysis of deficit round robin scheduling for future aeronautical data link," in *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, PIMRC*, 2011, pp. 1809–1814.

[17] G. Armitage and L. Stewart, "Some thoughts on emulating jitter for user experience trials," in *Proceedings of the ACM SIGCOMM 2004 Workshops*, 2004, pp. 157–160.

[18] M. Ehammer, T. Gräupl, and C. H. Rokitansky, "TCP/IP over aeronautical communication systems - Effects on bandwidth consumption," in *AIAA/IEEE Digital Avionics Systems Conference - Proceedings*, 2009.

Email Addresses

thomas.graeupl@dlr.de

mamayr@cosy.sbg.ac.at