Generic Access Network Emulation for NGN Testbeds

[Invited Paper]

Joachim Fabini*
TU Wien, Institute of
Broadband Communications
Favoritenstr. 9/E388
A-1040 Vienna, Austria
joachim.fabini@tuwien.ac.at

Peter Reichl
Telecommunications Research
Center Vienna (ftw.)
Donau-City-Str. 1
A-1220 Vienna, Austria
reichl@ftw.at

Christoph Egger
TU Wien, Institute of
Broadband Communications
Favoritenstr. 9/E388
A-1040 Vienna, Austria
christoph.egger@tuwien.ac.at

Marco Happenhofer†
TU Wien, Institute of
Broadband Communications
Favoritenstr. 9/E388
A-1040 Vienna, Austria

Michael Hirschbichler‡
TU Wien, Institute of
Broadband Communications
Favoritenstr. 9/E388
A-1040 Vienna, Austria

Lukas Wallentin§
TU Wien, Institute of
Broadband Communications
Favoritenstr. 9/E388
A-1040 Vienna, Austria

ABSTRACT

Evaluating the user satisfaction with Next Generation Network applications requires, amongst others, to carefully study the impact of various access networks with differing performance characteristics. This paper presents the concept and implementation of a network emulator for IP-based packet-switched wired and wireless access technologies based on measurement results obtained in real access networks. We present a generic modeling process which consists of three stages: (1) metric and measurement methodology selection, (2) measurements in real access networks, and (3) access network emulation based on the measurement results. Whereas the proposed modeling concept is generic and can be applied to any metric, the implementation part of this paper focuses on changes to the Open Source Linux WAN emulator NetEm which are required to accurately emulate one-way delay for access networks. We propose two distinct emulation options and compare one-way delay measurement results for real access technologies against the emulated ones, emphasizing limitations and pitfalls related to access network emulation in general.

Categories and Subject Descriptors

C.2.1 [**Computer**]: Communication Networks—*Network Architecture and Design*; C.4 [**Computer**]: Performance of Systems; I.6.5 [**Computing Methodologies**]: Simulation and Modeling—*Model Development*

General Terms

Measurement, Performance, Verification, Design

Keywords

IMS, NGN, Emulator, Access Network, UMTS, 3G

1. INTRODUCTION

After an extensive period of conceptual preparation, packet-switched Next Generation Networks (NGN) are currently becoming reality, represented by two main candidate architectures: the IP Multimedia Subsystem (IMS) standardized by 3GPP and the NGN architecture standardized by ETSI/TISPAN. The decision to publish IMS and NGN specifications free of charge has stimulated the development of several Open Source NGN activities, most notably the Open Source IMS project [6] due to Fraunhofer FOKUS which is supported by a huge worldwide community of users and developers.

Given the enormous technical excellence and market penetration of current 2G and 3G mobile networks, success or failure of Next Generation Networks will depend primarily on their ability to provide satisfactory user experience. Therefore, developers will have to test future NGN applications with respect to user acceptance, including underlying middleware, signaling, and media protocols. As NGNs are supposed to be access agnostic, this in particular will require to carefully study the impact of various packet-switched wireless and wired access technologies, with significantly differing performance parameters like delay, jitter or loss rate.

In practice, however, 3G access networks for Open Source IMS testbeds are commonly replaced by wired Ethernet links mainly due to cost considerations. As far as evaluations are concerned, this may lead to a significant decrease of the

*Corresponding author, email:Joachim.Fabini@tuwien.ac.at
†email: marco.happenhofer@tuwien.ac.at
‡email: michael.hirschbichler@tuwien.ac.at
§email: lukas.wallentin@tuwien.ac.at

TRIDENTCOM 2008, 17th – 20th Mar 2008, Innsbruck, Austria.

DOI 10.4108/icst.tridentcom.2008.3135

level of confidence in measurement results. Therefore, in this paper we present our concept and implementation for a measurement-based access network emulation which can be used as a highly accurate replacement for any current and future IMS access network. Following [1] and [2], we use a three-step approach for access network modeling, starting with a **metric selection** process based on user requirements which are matched against the IETF IP Performance Metrics (IPPM) framework [4]. During this process we end up with a set of metrics which is capable to capture the targeted access network behavior at a sufficiently high level of accuracy. This is used as an input to the **access network measurement** process which implements methodologies to measure parameter values (or "singletons" in IPPM terminology) for the chosen metrics in real networks. The measurement methodologies should be sufficiently generic in nature so that identical metric assessment procedures can be applied to a series of existing physical access networks. The resulting measurement data are pre-processed to generate either empirical data (e.g., tables containing delay as a function of packet size) or analytical models. Finally, appropriate **access network emulators (or simulators)** use the resulting empirical configuration data or analytical models for delaying packets on specific links. Ideally, access network emulators (or simulators) are inserted transparently into the testing path, connected to all systems under test through low-delay links. A typical use-case which we will present later on is a bridge connected by means of Ethernet wires into the testing path.

The remainder of this paper is structured as follows: Section 2 introduces basic concepts of metric selection using the IPPM model and presents our measurement methodology for the specific metric "one-way delay". For this metric, section 3 illustrates the measurement process using results from a live UMTS network. Based on these measurement results, section 4 presents two distinct access network emulation alternatives which we have implemented for an existing Open Source emulator, before section 5 discusses limitations which are common to payload-dependent delay emulators. Section 6 concludes the paper with a brief summary and outlook.

2. METRIC SELECTION AND MEASUREMENT METHODOLOGY

Selecting suitable metrics is basically a matter of two dominating aspects, i.e. the pronounced asymmetrical nature of many wired and wireless access networks, and the asymmetrical structure of typical client-server-based application-layer protocols, where "protocol symmetry" refers to both request-to-reply message count and message size ratios being close to 1. While, e.g., round-trip delay (RTD) might be a valid metric for the delay of highly symmetrical protocols over a specific wireless link, asymmetrical higher-layer protocols increase the complexity of simulation or emulation models because of their varying message size and varying ratio for requests and replies. Thus, for instance the delay between sending a Session Initiation Protocol (SIP) request and receiving the corresponding reply over an asymmetrical wireless link depends significantly on the sizes of both request and reply, as well as on whether the request was sent in the uplink and the response was received in the downlink or vice versa. This is particularly true in the case of huge presence- or geo-information XML bodies sent over the uplink as response to small SIP request messages which have been received in the downlink.

Summarizing, the focus of metrics selection for access network measurements is on capturing the asymmetric nature and behavior of these networks. In the remainder of this paper we follow the reasoning of the IPPM framework, which advocates a clean separation between metric, methodology, and measurement process.

2.1 IP Performance Metrics Model

The Internet Protocol Performance Metrics (IPPM) working group of the Internet Engineering Task Force (IETF) has standardized the base framework for IP Performance Metrics as RFC 2330 in September 1999. This framework defines requirements on performance metrics, recommends procedures and details on common uncertainty and error sources in IP network measurements. Several RFCs refine metrics based on the requirements of RFC 2330, the following ones being relevant for access modeling: RFC 2679 and RFC 3432 define one-way delay metrics, which are used by RFC 3393 to derive IP delay variation metrics. RFC 2680 proposes the corresponding one-way loss metric, which is refined by RFC 3357 to define loss patterns, whereas RFC 2681 introduces a round-trip delay metric for IPPM. Finally, RFC 3148 defines a framework for bulk throughput capacity measurements.

In the rest of this paper we illustrate the stages of our access network modeling process for the metric one-way delay as defined in RFC 2679. However, we would like to emphasize that similar methodologies, measurement and emulation processes can be defined for any other metric.

2.2 Measurement Methodology

High one-way link delay is one prominent characteristic of many access networks, specifically of 2.5G/3G cellular access networks. While one-way delays in lightly loaded wired local area networks and area-restricted core networks range typically in the order of tens to hundreds of microseconds, the delay in 2.5G and 3G networks can exceed this value by at least three orders of magnitude. Moreover, test measurements have indicated these wireless links to exhibit significant correlation between delay and packet payload size.

One of the requirements on our delay measurement tools is to bypass firewalls, NATs and gateways. Concerning the measurement methodology, this implies to select ICMP (Internet Control Message Protocol) echo as packet type for all delay measurements, due to several benefits: First, most firewalls and NATs permit ICMP echo messages to pass through, even if other protocols are blocked. Second, the request-response messaging pattern matches closely the signaling behavior that we target for assessing Session Initiation Protocol (SIP) behavior. Third, the ICMP echo message header size of 8 bytes is identical to the one of UDP messages, affording inference on the behavior of, e.g., RTP or SIP over UDP messages for specific payload sizes.

One notable downside of ICMP echo packets with respect to deterministic delay measurements is the fact that some network elements (routers) may treat ICMP packets differently

from TCP or UDP traffic. Nonetheless, for the relatively high delay that we expect to measure for access networks we argue that the delay difference between UDP, TCP and ICMP packets in core network elements is negligible.

2.3 One-way Delay Measurement Setup

According to RFC 2679, clock synchronization between two measurement hosts is the main uncertainty factor in one-way delay measurements. Therefore the clocks of these hosts are frequently synchronized against global time, e.g., using the Global Positioning System (GPS) as a common, accurate timebase, resulting in medium investments and configuration effort. However, we argue that clock synchronization is not required for accurate one-way delay measurements. The simple but highly accurate one-way delay measurement setup which we propose in the following uses **clock correlation** instead of **active clock synchronization**. The algorithm relies on time correlation markers which are used during offline correlation to compute accurate clock offset between server clock and client clock. The setup requires an extra, highly deterministic, low-delay network path – e.g., an Ethernet link, as well as an additional network interface for time correlation, both in the mobile terminal and in the measurement server. Figure 1 depicts the measurement setup which we have used for one-way delay measurements.

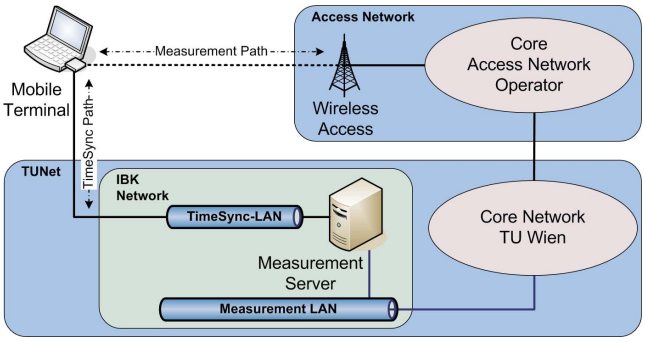

Figure 1: Generic Measurement Setup for Clock Synchronization and -Correlation

This setup will typically not require any extra hardware investments, nor does it necessitate any special drivers. However, the TimeSync path's round-trip delay is recommended to undercut the one of the path under test by orders of magnitude. Accuracy tests have indicated that accuracy of this time correlation setup is typically better than 0.05 ms for a switched Ethernet link, improving to values below 0.01 ms for Ethernet over a single crossed patch cable, depending on the client's and server's specific hardware and operating system.

In addition to our targeted singleton metric "Type-P-One-way-Delay" as defined by RFC 2679, our measurement tool also uses a sampling procedure resembling the sample metric "Type-P-One-way-Delay-Poisson-Stream" conforming to the same RFC. In contrast to RFC 2679, our tool samples one-way delay singletons at uniformly distributed times while RFC 2679 requires sampling times according to a pseudo-Poisson process. However, we consider uniformly distributed times to be sufficiently uncorrelated to not cause interference in the access link or in the core network. An additional extension to RFC 2679 is that our measurement methodology uses a second level of randomness, i.e. randomly distributed payload sizes, thus guaranteeing that temporary network overload situations impact on the delay of a broad range of payload sizes and not just on one payload or a few adjacent ones.

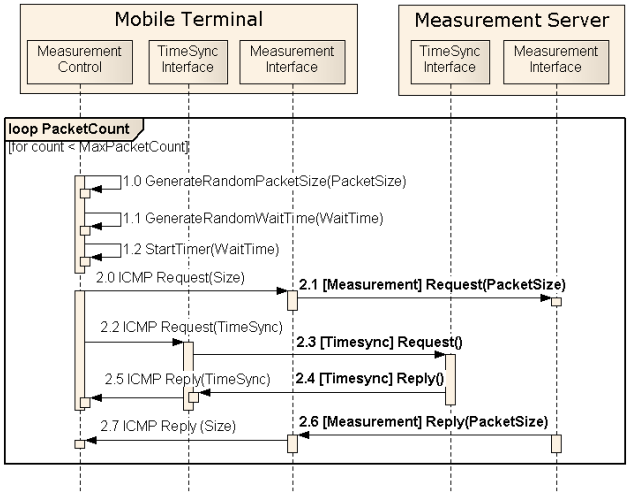

Figure 2: Sequence diagram of random payload one-way delay measurement script

Our measurement methodology relies on sending ICMP echo request pairs, one of which, having a small and constant payload size, uses the timesync path, whereas the other one, the actual random payload measurement probe, uses the measurement path. Fundamental to the tool's operation is that tcpdump [5] instances listen on all client and server interfaces and record timestamps along with headers of all sent and received ICMP messages. Figure 2 depicts the sequence diagram for random payload one-way delay measurements. For clarity reasons we have omitted startup and close-down procedures which start and stop tcpdumps on client and server using SSH-secured remote connections.

The random one-way delay loop in Figure 2 iterates over the user-configured sample count, i.e., it terminates after having sent a pre-configured total number of ICMP measurement probes. As first step within the random payload one-way delay loop the tool generates two random values: one for the packet payload size (message 1.0) and one for the wait time (message 1.1). Both values, the payload size value and the waiting time value are uniformly distributed within user-configured limits. Lower limit for ICMP payload size is 13 bytes to store the sending timestamp for round-trip delay calculation, whereas we restricted the upper payload size boundary to the Ethernet MTU (1472 bytes ICMP payload). Concerning waiting times, inter-packet-gap delays have to be chosen in such a way that the load generated by subsequent packets does not exceed the access link's maximum throughput capacity. For UMTS we have selected random waiting times between 500 ms and 1500 ms.

Prior to sending the measurement ICMP request the measurement tool starts a timer (message 1.2) to trigger the

next loop iteration – possibly before arrival of the ICMP reply to the current request. Finally, the tool sends the measurement ICMP request (messages 2.0 and 2.1) whose payload size matches the previously computed value and concurrently the timesync ICMP request (messages 2.2 and 2.3). The measurement server records arrival of these ICMP requests and sends timesync and measurement ICMP echo replies to the mobile terminal (messages 2.4, 2.5 and messages 2.6, 2.7, respectively) having a payload which matches the payload of the corresponding originating ICMP request. The departure of all ICMP replies is also recorded by the measurement server and their arrival by the mobile terminal. After measurement completion the offline correlation process computes one-way delays and losses for any measurement ICMP packet using the clock offset information which can be extracted from the timestamps of the closest ICMP timesync packet exchange.

3. MEASUREMENT RESULTS

In terms of performance, the 3G UMTS standard is positioned in between the 2.5G standards GPRS and EDGE and advanced 3G standards like HSDPA and HSUPA. Therefore, this section presents performance measurement results for UMTS Frequency Division Duplex (FDD) as the reference technology against which other measured wireless and wired technologies are to be compared.

3.1 Measurement Configuration

The mobile network we have assessed is *mobilkom austria*'s 3GPP Rel99 compliant live UMTS FDD network, offering maximum theoretical transmission rates of 64 kbit/s uplink and 384 kbit/s downlink. As UMTS modem we have used an Option Wireless UMTS PCMCIA card, integrated into Linux using usbserial drivers. Our mobile client was a Compaq Armada M700 laptop, equipped with a PIII 850MHz CPU, 576 MByte RAM and running SuSE Linux 10.2, whereas the measurement server was a COTS PC based on a 2GHz Pentium IV CPU equipped with 1 GByte of RAM and running also SuSE Linux 10.2. We have measured UMTS uplink and downlink one-way delay for IPv4 ICMP packets, configuring the loss limit which discriminates between losses and delay singletons according to RFC 2679 to be 5 seconds. We have sent a total of 10000 singletons.

3.2 Random Payload One-way Delay

Round-trip delay measurements lack information regarding the contribution of uplink delay and downlink to the total round-trip delay, as well as the IP delay variation contribution of uplink and downlink. Therefore, according to section 2, we measure one-way delay by dividing any round-trip delay singleton into one one-way uplink delay singleton and one downlink delay singleton. While round-trip delay singleton values have been acquired using the ping utilities' output, one-way delay singletons are recorded using correlated tcpdump trace files on the measurement server and on the mobile terminal.

Figure 3 depicts the UMTS network's uplink one-way delay profile. The diagram shows packets having adjacent payload sizes within blocks of 160 bytes to be subject to equal delays and identical IP delay variation. Note that the rightmost (full 160 bytes) block shown in the diagram is delayed by

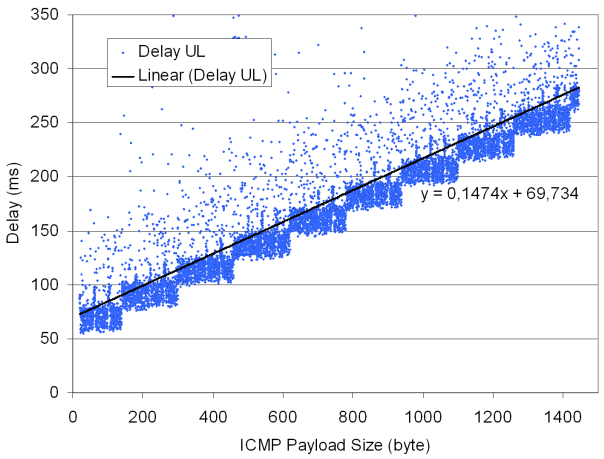

Figure 3: UMTS uplink one-way delay (random inter-packet delay and random payload size)

a median value of 250 ms, which is 3.5 times the leftmost block's delay of 70 ms. Additionally, we can observe a huge number of outliers which cause a vertical offset to the linear interpolation line.

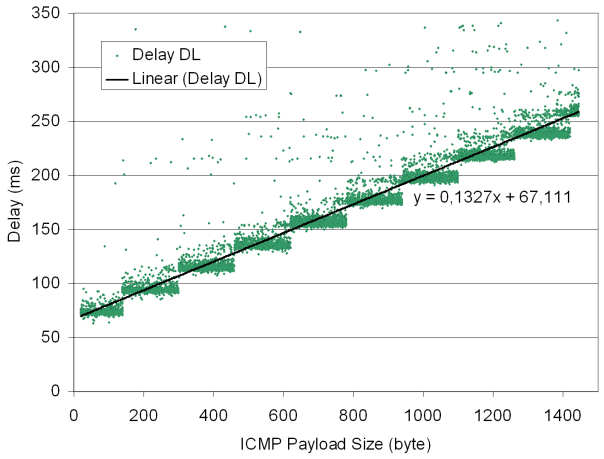

Figure 4: UMTS downlink one-way delay (random inter-packet delay and random payload size)

Comparing the UMTS uplink delay (Figure 3) against the UMTS downlink delay (Figure 4) we infer that the borders of downlink delay blocks correlate with the uplink ones. Also the initial delay (for small payload sizes) and the increase in delay with payload for UMTS downlink is almost identical to UMTS uplink.

However, from Figure 4 we conclude that the UMTS downlink is subject to much less outliers than the UMTS uplink. Moreover, the IP delay variation (IPDV) for UMTS downlink blocks amounts to roughly half of the UMTS uplink's IPDV. This finding is confirmed by comparing the delay histograms for uplink (Figure 5) and for downlink (Figure 6). While the uplink histogram shows a relatively flat curve, downlink singletons are grouped into 10 narrow spikes, any

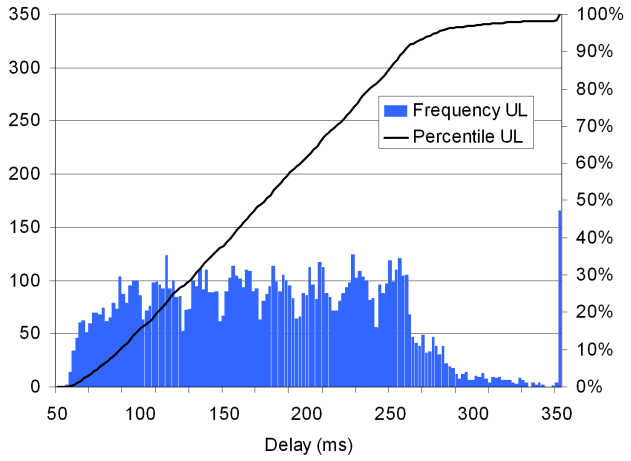

Figure 5: UMTS uplink one-way delay histogram (random inter-packet delay, random payload size)

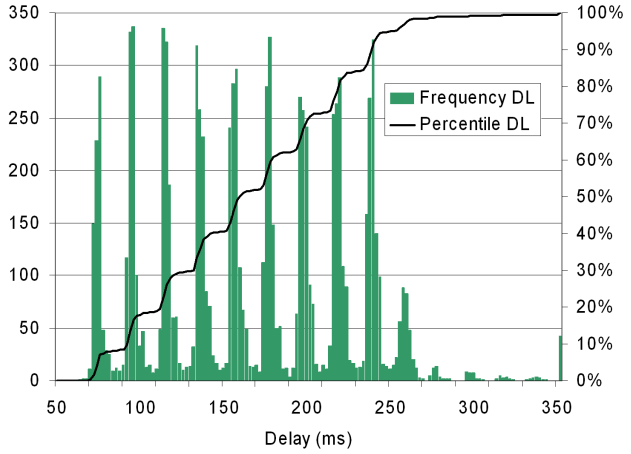

Figure 6: UMTS downlink one-way delay histogram (random inter-packet delay, random payload size)

spike corresponding to one of the 10 blocks in the downlink delay diagram (Figure 4, note that the rightmost block, which corresponds to the largest tested payload sizes, is incomplete and therefore the rightmost spike is smaller than the other ones).

4. ACCESS NETWORK EMULATION

As representatives of the third and last stage of our model, in this section we present two distinct approaches to real-time access network emulation, discuss implementation changes and compare measurement results of our implementation against the reference measurements in real access networks. Our access network implementation relies on the NetEm Linux WAN emulator module.

4.1 NetEm functionality

NetEm [3] is an Open-Source module which acts as a queuing discipline, being integrated into the Linux 2.6 kernel sources. NetEm's command-line interface uses the tc utility which is part of the IPRoute2 tools. NetEm features emulation of various impairment parameters like constant delay, statistical jitter, statistical loss, and reordering on a per-interface base. For further details on NetEm's architecture and functionality we refer to [3].

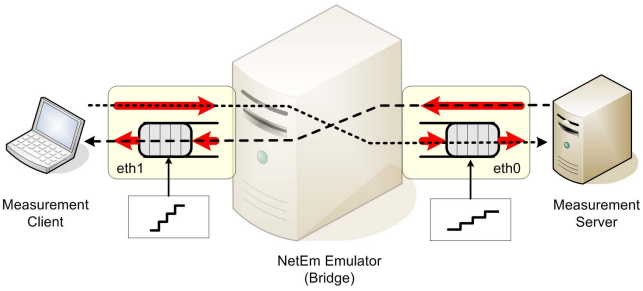

Figure 7: NetEm WAN Emulator concept

Figure 7 depicts NetEm's basic queuing concept. Packets received on an incoming interface are passed to NetEm which processes the packets according to the configured impairment factors before enqueuing them on the outgoing interface. Therefore, as indicated in the figure by means of distinct impairment function symbols, NetEm impairment parameters can be set on a per-interface base, which is a pre-requisite for modeling unidirectional links. E.g., in Figure 7 the interface *eth*0 along with its associated NetEm impairment factors models the access network's uplink, while being fully transparent for downlink packets. For the reverse direction, NetEm impairment parameters associated with Ethernet interface *eth*1 model the downlink, acting exclusively on packets sent by the measurement server toward the measurement client.

The precision of NetEm depends on Linux kernel configuration parameters (specifically the kernel tick Hz value), although, starting with kernel release 2.6.22, NetEm also supports sub-kernel-tick delay resolution. However, NetEm operating as part of a standard Linux 2.6.20 kernel compiled for 1 ms kernel tick is sufficiently accurate to be used for the targeted access network emulation.

For accurate and transparent access network emulation, the Linux server hosting NetEm in Figure 7 was set up as a bridge using the Linux Net:Bridge module, being fully transparent at IP level. The measurement client is connected to one of the bridge's interfaces, the measurement server to another bridge interface using 100 Mbit/s Ethernet devices.

4.2 Linear Delay Emulation

With respect to access network emulation, one major deficiency of NetEm is its inability to correlate impairment parameters with packet payload size. Therefore we have implemented packet-size-dependent delay using linear delay approximation as a function of packet size. However, even if our current implementation focuses on payload-dependent delay, we would like to emphasize that the described emulation concept is generic in nature and that an implementation procedure analogous to the one presented below can be used to easily integrate other payload-dependent metrics like, e.g., payload-dependent loss with NetEm.

To this end, we have extended NetEm's tc command line interface to accept an additional command line keyword "slope" followed by a mandatory argument which represents the increase in delay per payload byte. When used in conjunction with the slope argument, NetEm's existing constant delay parameter equals the delay offset for a payload size of zero, delay(0). This value corresponds to the intersection of the delay curve's linear approximation with the y-axis in the diagram in Figure 3 for uplink and Figure 4 for downlink, while the value-argument to the slope parameter represents the inclination of the respective size-delay curve. An incoming packet having a payload value of *size* is therefore delayed by an amount $delay(size) = delay(0) + slope * size$. Following NetEm's internal processing chain, this effective payload-dependent delay value is then adjusted subject to a uniformly distributed IP delay variation whose mean value also depends on the specific link measurement results. This second stage of processing adds the level of uncertainty to the delay which is required for accurate access network emulation.

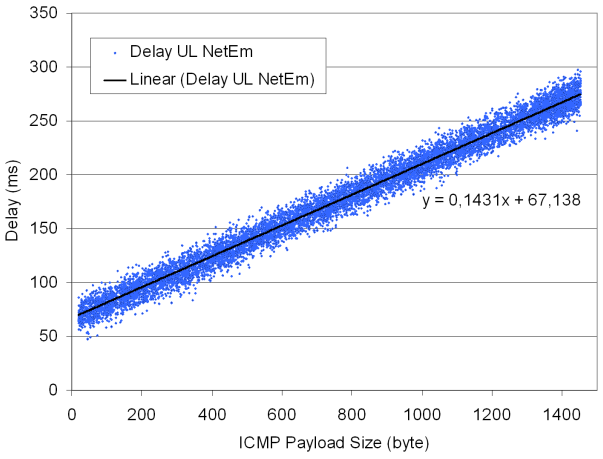

Figure 8: UMTS NetEm linear uplink delay emulation results

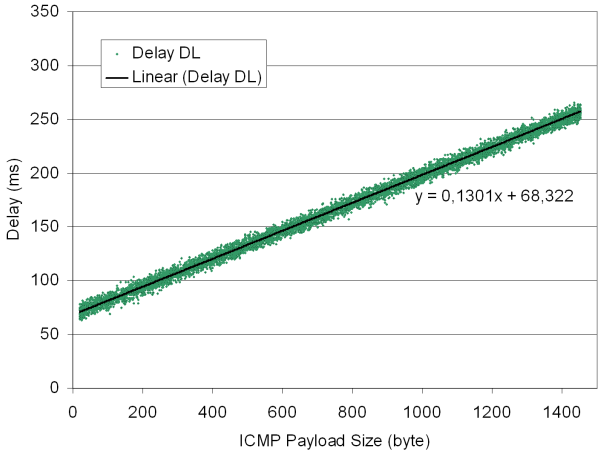

Figure 9: UMTS NetEm linear downlink delay emulation results

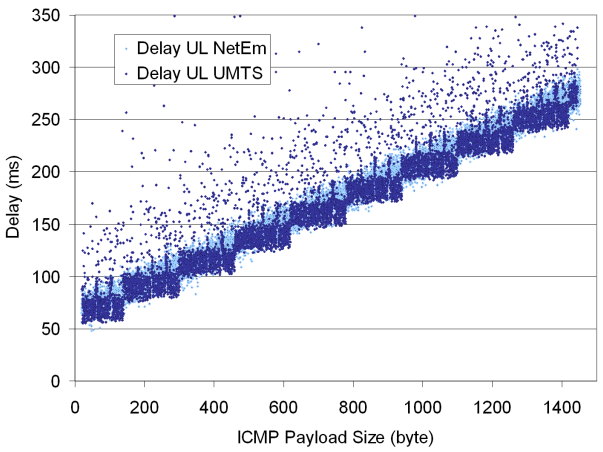

Figure 10: UMTS NetEm linear uplink delay correlation with UMTS results

Figure 8 and Figure 9 depict uplink and downlink one-way delay measurement results for linear NetEm UMTS emulation. We have configured NetEm using the slope and initial delay values shown in Figure 3 and Figure 4 configuring an IPDV (NetEm jitter) value of 10 ms for uplink and 4 ms for downlink.

An important measure of emulation quality is the correlation between original measurement data and the corresponding linearly interpolated emulation results. Figure 10 compares these two measurement results for UMTS uplink. The diagram indicates that the linear emulation's slope and jitter parameters have been configured correctly, whereas the linear emulation fails to reproduce the huge number of outliers which are present in the original UMTS uplink delay measurements. In addition the emulation generates some delay values which do not match the original ones (represented by lightly colored dots visible close to the blue UMTS blocks' edges) while other original UMTS delay values are not generated by the emulation (specifically the lower corner of the UMTS blocks).

Primary reason for missing outliers is NetEm's normal jitter distribution which we have used. If required for specific tasks, replacing the normal distribution by a heavily right-tailed distribution can generate these outliers. However, the linear emulation for round-trip delay is in line with our requirements to emulate "typical" UMTS behavior.

4.3 Table-based Delay Emulation

Linear approximation can provide satisfactory accuracy for most access technologies. However, the linear interpolation for UMTS uplink in Figure 8 and Figure 9 fail to account for the discrete increase in delay of almost 20 ms median value at the edges of the rectangle blocks. An ideal linear interpolation (crossing the block centers) deviates by -10 ms for the lower payload value and by $+10$ ms for the higher payload value when compared to the measured median value.

Therefore we have improved the slope-based emulation by implementing a table-based access network emulation which

maps payload values to delay values. An important factor which finally decides on emulation accuracy is the algorithm used for converting access network measurement results to emulator delay tables. The solution which we have adopted is to use a 3-payload aggregated point-wise median as shown in Figure 12 and Figure 13 for uplink and downlink, respectively. The delay table value for a payload size of s is computed as the median delay value for all measurement values of payloads $s-1$, s, and $s+1$.

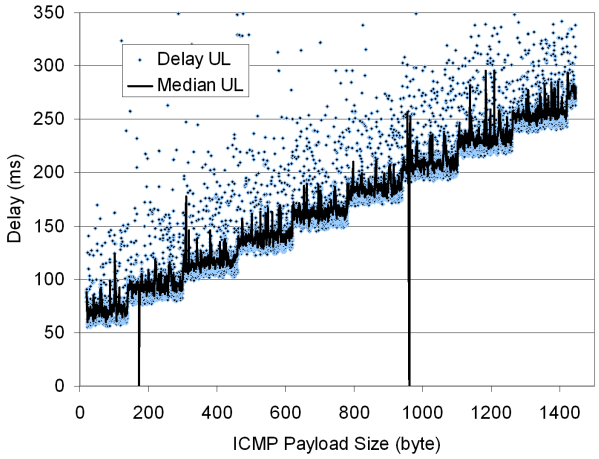

Figure 11: UMTS uplink delay curve based on payload-size point-wise median

Subject to a sufficiently high number of singletons, the most accurate delay representation yields a curve which consists of the median delay value computed separately for any payload size. However, the shape of the median curve for uplink in Figure 11 illustrates that randomly chosen payload values combined with a too low number of singletons can lead to missing delay singletons for specific payload sizes. The median delay curve indicates missing singletons for two payload values (173 bytes and 959 bytes), as well as significant outliers for other payload sizes.

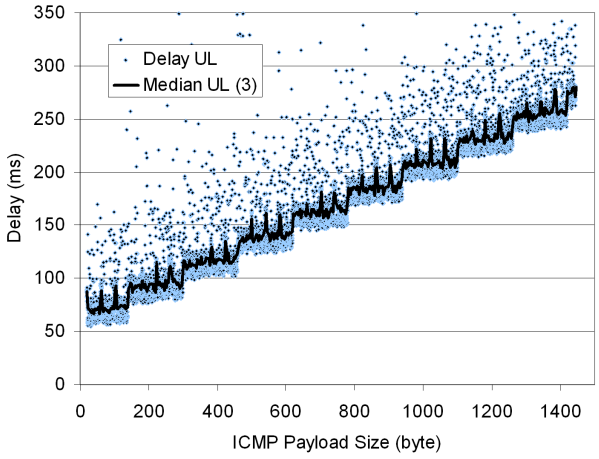

Figure 12: UMTS uplink delay curve based on 3-payload-size aggregated median

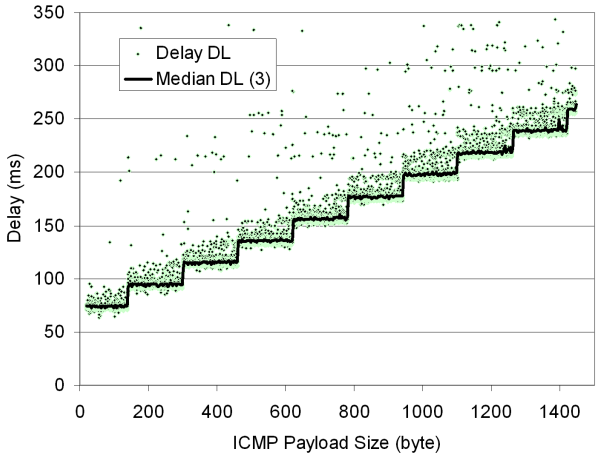

Figure 13: UMTS downlink delay curve based on 3-payload-size aggregated median

One solution to eliminate this uncertainty factor is to increase the number of measurement singletons. However, increasing the singleton count results in higher measurement costs, as well as significantly increased overhead associated with one-way delay singleton correlation. Therefore we have decided to adopt an alternative solution which consists of aggregating groups of delay singletons for adjacent payload sizes. Our evaluation has lead to the conclusion that for the selected number of 10000 singletons distributed randomly over approximatively 1450 payload sizes the median curve aggregating 3 adjacent payload sizes, as shown in Figure 12 is the best tradeoff between accuracy and outlier avoidance.

Our implementation of table-based delay re-uses the concept which NetEm has adopted for jitter implementation. We have added a new command-line argument `delaytab` to the `tc` command line which expects two mandatory parameters: one table name (e.g., `umts`) and one direction parameter (e.g., `UL` for uplink or `DL` for downlink).

The table consists of lines which store space-separated pairs of MAC-layer payload size and corresponding delay value. The tc utility reads the table, converts and transfers it to the kernel space, where one such table can be instantiated per physical or logical network interface. Once NetEm was configured to use the table, it delays any incoming packet based on its payload size. NetEm's native implementation of delay and jitter parameters was not affected by the delaytab parameter, therefore table-based delays can be subject to IP delay variation, distribution and correlation like any constant NetEm delay values.

Whereas the emulation – similar to linear slope based emulation – fails to emulate the huge number of outliers, a comparison of the two diagrams shows that the typical case is perfectly emulated in terms of delay and IP delay variation. Specifically the narrow spikes visible in the uplink delay diagram in Figure 3, caused by hardware anomalies in the original UMTS measurement setup's modem, are emulated correctly. Adding appropriate outliers is feasible by replacing NetEm's normal delay distribution by an appro-

priate distribution. However, we consider it important to focus on typical emulation delay aspects and therefore we will postpone the outlier distribution for further study.

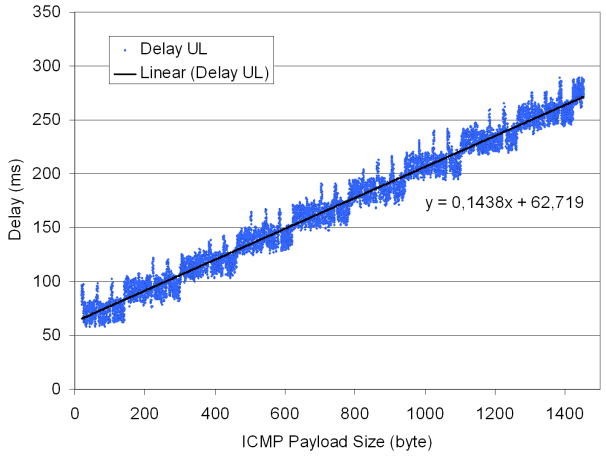

Figure 14: UMTS NetEm table-based uplink delay emulation results

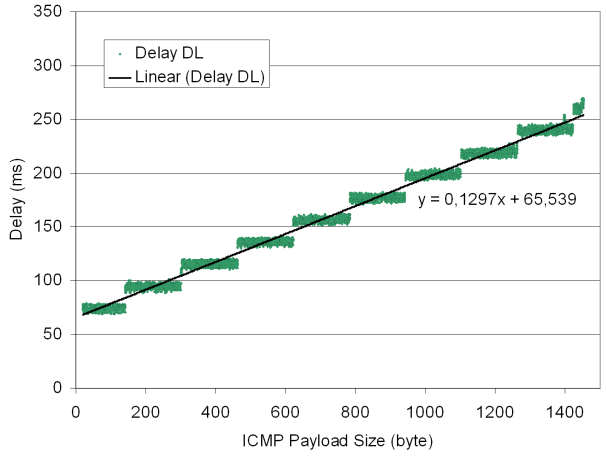

Figure 15: UMTS NetEm table-based downlink delay emulation results

An analysis of table-based emulation uplink and downlink measurement results (depicted in Figure 14 and Figure 15, respectively) confirms that the above-mentioned delay spikes, as required, impact exclusively on the uplink's one-way delay. Moreover, conforming to the original UMTS measurement result values shown in Figure 3 and Figure 4, the emulation's IP delay variation value for uplink is almost double of the downlink's IPDV value. The IPDV value can be mapped graphically to the height of rectangle blocks in the above-mentioned diagrams which averages a value of 25 ms for uplink blocks and 12 ms for downlink blocks.

4.4 Emulation Performance

Preliminary performance tests running our modified NetEm implementation on a Pentium III 850 MHz PC, equipped with 512 Mbyte of RAM and running SuSE Linux l0.2 indicate that our implementation changes do not deteriorate

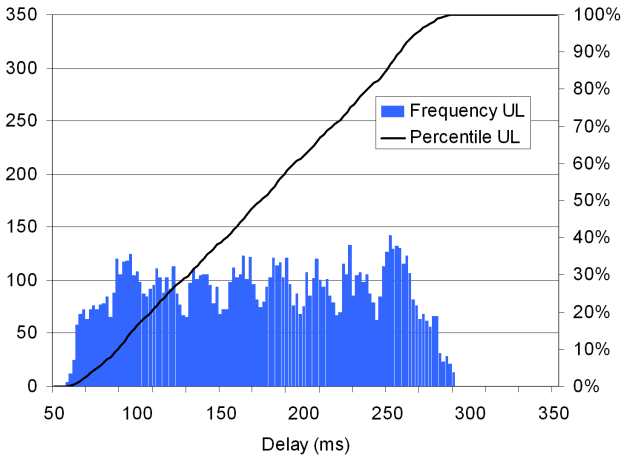

Figure 16: UMTS NetEm table-based uplink delay emulation histogram

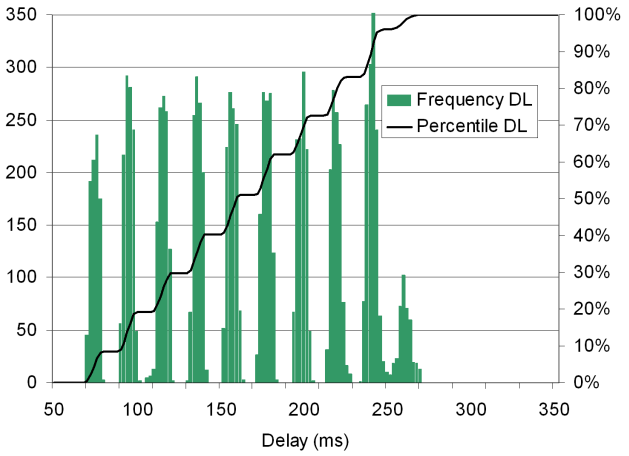

Figure 17: UMTS NetEm table-based downlink delay emulation histogram

NetEm's original good performance. Configured as a bridge, the emulator is capable to handle a limit of approximatively 150000 packets per second, being limited by interrupt handling. Using two 100 Mbit/s Ethernet interfaces the CPU was only lightly loaded (10-15%), only after replacing the interfaces by Gbit Ethernet and increasing the MTU we were able to fully load the CPU. However, targeting transfer rate limited access networks the emulator is capable to accurately simulate tens to hundreds of concurrent links, depending on the specific access technology and configuration which it emulates.

5. LIMITATIONS OF ACCESS NETWORK EMULATION

The previous sections have illustrated how access network delay can be emulated using linear and/or table-based emulation methodologies. However, there are some limitations which must be considered for realistic access network emulation. It is important to note that these limitations are not

specific to our emulator implementation, but rather apply to any IP-level emulator which enforces payload-size depending delay.

5.1 IP Fragmentation

The first important aspect concerns IP fragmentation. The enhanced access network emulator is configured as a bridge and can therefore be connected transparently in between the measurement server and the measurement client using two 100Base TP Ethernet interfaces. All IP packets bypassing the emulator are fragmented based on the path MTU size, meaning that packets exceeding the path MTU size are divided into fragments having a size which is less than or equal to the Ethernet MTU of 1500 bytes. SIP signaling messages can easily exceed this limit, particularly SIP messages including location or presence XML documents can equal sizes of 5 KBytes or more.

Without dedicated fragment support, an emulator delays any separate IP fragment based on its fragment size and not based on the size of the original IP packet. E.g., assuming an Ethernet path MTU of 1500 bytes, an IP packet of 3500 bytes size (including IP headers) is fragmented into three parts, two 1500 bytes fragments and one 540 bytes fragment. In the linear emulation case the expected emulation delay is $delay = slope * 3500 + delay_0$ whereas the effectively emulated delay due to fragmentation will be $delay = slope * 1500 + delay_0$, as all three fragments will arrive within a short timeframe. Even more severe, the third fragment of 540 bytes time will overtake the larger fragments, causing a reordering situation.

The solution we have adopted is based on the methodology which the IP protocol itself uses for re-assembling IP fragments. Part of any IP header is a field termed fragmentOffset which stores the offset of the current fragment (in bytes) relative to the start of the IP packet. Delaying any single fragment by an amount fragmentOffset + fragmentSize safeguards that the delay emulated for the last fragment equals the delay of the entire IP packet. The receiver's IP stack is required to wait for IP packet re-assembling until the last fragment has arrived, therefore assuring that the emulator generates correct delay. This methodology is appropriate and provides reliable results for both, the table-based and for the linear emulation approach.

5.2 FIFO Queuing Aspects

One aspect of size-based delay emulation which we have mentioned in the previous subsection, namely small IP fragments overtaking large ones, is not restricted to IP fragments but also applicable to IP packets. Particularly whenever two IP packets arrive shortly one after each other at an emulator's input port and the first packet's payload size significantly exceeds the one of the second packet, chances are high that the emulator will send out the second packet prior to sending out the first one. The reason for this re-ordering is the way NetEm emulates link delay. Indeed, NetEm computes a delay value for any incoming packet, converts this delay to a desired output (dequeue) time and then places the packet in the outgoing queue. Therefore, once size-based delay is implemented, larger packets will have a later output timestamp than smaller packets, risking to be overtaken by smaller ones which have arrived short time after the large ones. The higher the emulator's configured slope factor, i.e., the more limited the access network's transfer capacity, the more pronounced the reordering effect will be.

However, this emulator behavior is not observed in real access networks where the network delay is caused primarily by limited access network transfer capacity and where the packet ordering is maintained. I.e., real access links exhibit strict FIFO ordering. To maintain this FIFO ordering concept for delay-based emulation we have changed NetEm's queuing concept with respect to the output timestamp. Specifically we have added a "latest timestamp" marker to the outgoing queue which remembers the timestamp (as absolute time) when the last of all currently enqueued packets is due to leave the queue. Whenever enqueuing a new packet its output time is modified in that we add the queue's latest timestamp marker value to the packet's computed delay value. After increasing the queue's latest timestamp marker to the packet's output time the packet is enqueued.

The queuing methodology which we have implemented emulates the behavior of typical 3G access networks, where Packet Data Protocol (PDP) contexts enforce strict FIFO queues for user data.

5.3 Scheduling Effects and IP Delay Variation

Our measurements have shown that emulation of shared channels is much more challenging than emulation of dedicated channels. Primary reason is the access network's scheduling strategy for specific scenarios which highly depends on the cell load at a specific point in time. As an example we present the diagram showing downlink measurement result for Cell-DCH state HSDPA delay in Figure 18.

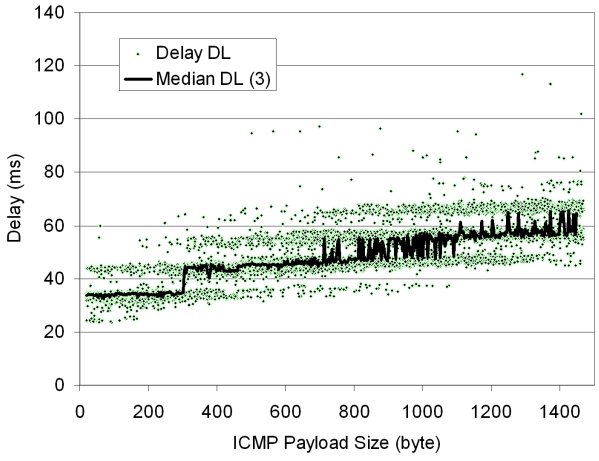

Figure 18: HSDPA random payload downlink delay for Cell-DCH state (3-payload median)

Supposedly because of scheduling effects on the Iub interface, the delay diagram in Figure 18 presents a noticeable layered structure for downlink singletons. The inter-layer distance of about 10 ms delay is an explicit indication that the HSDPA TTI of 2 ms can *n*ot be the reason for this layering. Therefore we consider the Iub interface with its 10 ms scheduling interval to be the most likely candidate. Depending on the specific cell/link load, the Iub scheduler spreads a

specific payload size over a distinct number of frames. What makes emulation so difficult is that our proposed methodology can not capture external load factors, which bias significantly on the measurement result as illustrated by the 3-payload download median curve in Figure 18.

A histogram analysis indicates that the second layer (centered at 57 ms delay) comprises about 50% of all downlink delay singletons while the third layer (centered at 67 ms) aggregates about 40% of the singletons. However, what makes emulation really challenging is that for payload sizes between 700 bytes and 1100 bytes the median curve oscillates between the second and the third layer. Our conclusion is that the median-based methodology which was appropriate for emulating most other access technologies must be enhanced for shared channel emulation.

We consider percentile-based probabilistic functions which select one delay value out of a discrete set of possible delay values as most promising solution for a realistic shared channel emulation. Applying NetEm's jitter and some improved outlier generation functionality on this delay value can help in generating the required layered delay structure. This functionality is not yet implemented but considered as a topic of highest priority for future Work.

5.4 Emulating Beyond Measurement Limits

The size of IP datagrams is limited to 65535 bytes. Even though it is unlikely to encounter such huge datagrams in real applications, emulators must be prepared to delay any IP packet up to the maximum IP datagram size. However, financial effort and measurement duration for acquiring a sufficiently large number of random delay singleton for this huge payload range can be substantial.

Therefore we have limited our access network measurements to payload sizes up to the Ethernet MTU of 1500 bytes. Test measurements for larger payload sizes confirm repeating delay patterns for tested 2G and 3G access technologies. Provided that the MTU of any network in the measurement path is smaller than the Ethernet MTU we consider it therefore legitimate to extrapolate measurement results for our specific payload range to a larger payload range. If the above-mentioned MTU condition is not satisfied, specifically, if the access network MTU is larger than the Ethernet MTU, additional measurements are required to avoid impact of larger MTUs on the measurement results. E.g., in the case of WiMAX, the round-trip delay of about 33 ms doubles when the payload exceeds the WiMAX MTU which was configured to be 1500 bytes.

While linear emulation works seamlessly for huge payload sizes, additional effort is required for extrapolating existing table-based measurement results for larger payload sizes. To this end, the measurement range's starting block and ending block must be correlated, taking into account the delay offset. However, automatic extrapolation is difficult and can become highly complex for unknown access technologies without a detailed analysis of layers below IP.

6. CONCLUSIONS AND FUTURE WORK

In this paper we have presented and discussed a generic approach to measurement-based access network modeling. The key advantages of our proposal focus on the simplicity of the model as well as its automated concept. Although metric selection and metric assessment implementation are two initial steps which require intense human interaction, once a measurement framework is implemented it can acquire data for any existing access network infrastructure. The resulting measurement data can be processed automatically to extract relevant parameters for simulation or emulation, except for the case of analytical modeling, which - if used at all - requires again human intervention. However, once the measurement parameters have been acquired for all access technologies under test, fully automated, client-controlled or server-controlled application testing can be implemented by means of the framework which we have presented.

The proposed framework is flexible and easily extensible, as new metrics can be added to the existing testing infrastructure by means of the control path loop mentioned above, while carefully dealing with potential side effects on the existing metrics. Some of the directions of current and future work have already been indicated in section 5 and are expected to further drive our proposal in becoming a major step towards a fully automated concept of modeling various access technologies for NGN networks.

7. ACKNOWLEDGMENTS

The authors would like to thank Dr. Werner Wiedermann and Dr. Günther Pospischil from mobilkom austria AG for facilitating the UMTS measurements and Stephen Hemminger for his work on NetEm and his support in adding the slope parameter. Part of this work has been funded by the Austrian Kplus competence center programme.

8. REFERENCES

[1] J. Fabini, P. Reichl, and A. Poropatich. A Generic Approach to Access Network Modeling for Next Generation Network Applications. *Proceedings of the 4th International Conference on Networking and Services (ICNS08), Gosier, Guadeloupe*, March 2008.

[2] J. Fabini, P. Reichl, and A. Poropatich. Measurement-Based Modeling of NGN Access Networks from an Application Perspective. *Proceedings of the 14th GI/ITG Conference on Measurement, Modeling, and Evaluation of Communication Systems (MMB 2008), Dortmund, Germany*, April 2008.

[3] S. Hemminger. Network Emulation with NetEm. *Proceedings of the 6th Australia's National Linux Conference (LCA2005), Canberra, Australia*, April 2005.

[4] Internet Engineering Task Force (IETF). *IP Performance Metrics (ippm)*, http://www.ietf.org/html.charters/ippm-charter.html, December 2007.

[5] Lawrence Berkeley National Laboratory, University of California, Berkeley, CA. *TCPDump Monitoring Tool*, http://www.tcpdump.org/, December 2007.

[6] D. Vingarzan and P. Weik. Development of an Open Source IMS Core for Emerging IMS Testbeds. *Journal on Mubile Multimedia (JMM), Special Issue on IMS*, 2(3), June 2007.