Delay Measurement Methodology Revisited: Time-Slotted Randomness Cancellation

Joachim Fabini and Michael Abmayer

***Abstract*—The operation of today's networked applications and protocols depends to a large extent on accurate end-to-end delay measurements and estimations. Appropriate methodologies can significantly improve quality, accuracy, and representativeness of delay samples acquired through measurements. This paper focuses on limitations of state-of-the-art end-to-end delay measurement methodologies. Its central observation is that the first time-slotted link in a measurement path cancels start-time randomness of delay measurement samples. When leaving this first link, all measurement samples are time-synchronized with each other and potentially with global time modulo link period. Because of this effect, which the paper introduces as time-slotted randomness cancellation effect, random sampling is not possible beyond the first time-slotted link of measurement paths. End-to-end measurements, therefore, fail to capture the full delay range of subsequent time-slotted links in the path, measurement results being limited to a specific session setup. Following a detailed discussion of theoretical models, the paper proposes a novel delay measurement methodology that adds artificial delay functionality to intermediate network nodes. Measurement packet headers include random seeds that are used by compatible ingress nodes of subsequent time-slotted network segments to regenerate start-time randomness. Samples acquired with this measurement methodology can assess a network path's full delay range. Practical applicability of the presented concept and methodology is demonstrated by a prototype implementation that assesses delay in public mobile cellular networks. Measurement results presented in this paper confirm that the proposed concept and methodology is generally applicable and that randomness regeneration can significantly improve quality and representativeness of delay measurement samples.**

***Index Terms*—3G mobile communication, delay effects, delay measurement methodology, IP networks, measurement errors.**

I. Introduction

DELAY measurements become increasingly indispensable for safeguarding acceptable user experience and network performance. Correct operation of network algorithms depends to a large extent on measurement-based delay approximations, estimations, and statistics. Concomitant, accurate round-trip delay (RTD) and one-way delay (OWD) measurements in today's networks become increasingly challenging because of overall network performance optimization and reactive network behavior. These optimizations introduce additional uncertainty parameters, e.g., cell load or total network load that influence on the determinism of end-to-end user-perspective delay measurements.

Standardization organizations and previous publications have isolated a series of fundamental measurement parameters as preconditions for accurate delay measurements. Among them, a paramount importance must be attributed to clock synchronization for OWD measurements [1] and start-time randomness as recommended by the IP performance metric (IPPM) framework [3].

However, this paper demonstrates that state-of-the-art end-to-end measurement methodologies are unable to capture a network's full delay range for specific setups even in deterministic conditions. Time-slotted network links in the measurement path can dramatically impair on measurement representativeness, limiting result validity to the respective measurement session. Moreover, when decomposing end-to-end delay measurement results into hop-by-hop delays, the delays of links following the first time-slotted network segment will exhibit multimodal delay distributions. Common statistics such as average or median, which assume normal distributed samples, can fail or at least provide questionable results when processing such multimodal distributed samples.

The novel delay measurement methodology, which this paper proposes, supports active measurements in acquiring representative, unbiased measurement samples in time-slotted network paths. Such unbiased measurement sample sets are an essential prerequisite for accurate end-to-end or hop-by-hop post-measurement analysis and processing. Being outside of this paper's scope, possible application areas include representative black-box or white-box assessment of link delay, as well as accurate evaluation of measurement methodology properties like, e.g., repeatability, continuity, or robustness.

A. Basic Terms and Concepts

Delay measurement methodologies can be categorized into passive measurements, which observe existing packets passively by means of probes in network nodes, and active measurements, which generate specific measurement traffic.

An orthogonal classification differentiates measurements depending on the level of access to the network path under observation into black-box and white-box measurements. Active end-to-end delay measurements are frequently conducted

Manuscript received September 19, 2012; revised January 22, 2013; accepted January 23, 2013. Date of publication July 9, 2013; date of current version September 11, 2013. The Associate Editor coordinating the review process was Dr. Dario Petri.

J. Fabini is with the Institute of Telecommunications, Vienna University of Technology, Vienna 1040, Austria (e-mail: joachim.fabini@tuwien.ac.at).

M. Abmayer is with Bahr IT Consult GmbH, Hinterbruehl 2371, Austria (e-mail: michael@abmayer.at).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIM.2013.2263914

in a black-box manner. This assumes that the network path under observation is unknown, restricting delay observation to two communication endpoints. Alternatively, white-box methodologies use detailed knowledge on network path internals to optimize measurements and/or collect information in intermediate nodes. Comparing the two approaches, black-box measurements are simpler to implement and less intrusive, whereas white-box measurements can offer a detailed hop-by-hop delay perspective at the cost of requiring access to intermediate network nodes and detailed knowledge of the network architecture. The latter two topics can be a serious security issue in live operational networks.

The novel solution that is proposed in this paper bases on active black-box measurements. It prevents disclosure of and/or access to intermediate network nodes by providing a set of random data and nodewritable information fields within each measurement packet header. Cooperative intermediate nodes within the network path can use this randomness improvement to support measurements without disclosing their identity, therefore, maintaining the high level of security and safety that is characteristic to black-box measurements.

B. Related Work

A large number of publications and standardization organizations focus on delay measurement, including the Internet Engineering Task Force (IETF) and International Telecommunication Union (ITU). A broad overview of standards, measurement technique concepts, as well as a state-of-the-art review of the related literature is presented in [1], in which the authors focus on OWD measurements and required clock synchronization techniques. Since the time of writing (2008), the cost of clock synchronization devices based on the global positioning system (GPS) with pulse per second (PPS) functionality has decreased significantly. PPS functionality has been seamlessly integrated with the Linux kernel and with the Network Time Protocol (NTP), such that application-transparent global time synchronization of Linux-based hosts down to 10 μs accuracy [8] involves costs of less than \$100 per unit. Detailed measurements conducted by the authors of [8] confirm the observation of [1] concerning PC hardware influence on measurements.

However, the authors of [1] omit to emphasize the importance of start-time randomness, which is one central component of the IETF's IPPM framework [3] and its associated metrics for OWD [4], RTD [5], and IP delay variation [6]. Measurements published earlier [9] and [10] and in this paper demonstrate that measurement methodology, i.e., randomized measurement sample start-time, can reveal – or hide – systematic OWD variations of tens of milliseconds in state-of-the art mobile cellular high-speed packet access (HSPA) networks. Following the definition of its framework [3], the IPPM group has defined protocols and architectures that standardize measurements for operational purpose, including definition of the one-way active measurement protocol (OWAMP [18]) and two-way active measurement protocol (TWAMP [19]).

Recently, the IPPM group has started an initiative that identifies roles and audiences for metrics and measurements. The resulting RFC 6073 [7] introduces the concept of point-of-view (PoV) on measurements, emphasizing the huge bias of parameters like, e.g., measurement interval, test stream characteristics, packet type, and sample size onto measurement results. The authors conclude that selection of appropriate measurement methodologies and statistics depends to a large extent on the specific application.

Numerous other publications consider active end-to-end delay measurements for mobile networks from various perspectives, ranging from conceptual, methodological, and network-centric point of view [11], [13]–[15] to signaling and application performance [16] and [17], to name just a few. The authors of [12] use passive measurements for dissecting end-to-end delay in operational mobile networks into its parts.

However, to the best of our knowledge, the randomness cancellation effect has not been presented and discussed so far, although the measurement results presented in this paper demonstrate its huge influence and impact on measurement results.

C. Structure of This Paper

The remainder of this paper is structured as follows. Section II presents the generalized problem statement of time-slotted randomness cancellation (II-A) and its particular application for RTD measurements in cellular networks (II-B) with time-slotted operation. Section II-C discusses various implications for delay measurements and recommends countermeasures. Theoretical concept and practical realization of the solution that is proposed to overcome randomness cancellation is outlined in Section III, followed by the measurement setup in Section IV, and measurement results in Section V. The paper concludes with a summary and outlook on future work in Section VI.

II. Problem Statement

Delay measurements require random send times to eliminate possible timing correlations between network and measurement packets. This applies particularly for time-slotted networks. This paper argues and demonstrates that this randomness condition is not fulfilled for the second and all following time-slotted network segments in a typical network path. As a main consequence, resulting measurement samples are limited in their representativeness and generality to the specific measurement session.

A. Generalized Case: Time-slotted Networks

Time slots in network segments are commonly allocated periodically on a per-session base. A session will have an opportunity to send data every k milliseconds, common period values being, e.g., 10 ms or 20 ms, depending on the specific network technology and configuration. The timing in these networks is high precision and global-time synchronized.

Accurate end-to-end delay measurements for network paths that include a time-slotted network segment will encounter a delay variation $d_x^w(i)$ of at least the time slotted network segment's period δ_x as depicted in Fig. 1. On constant upstream link delays $d_{s,x}(i)$ the arrival time of packet P(i) at

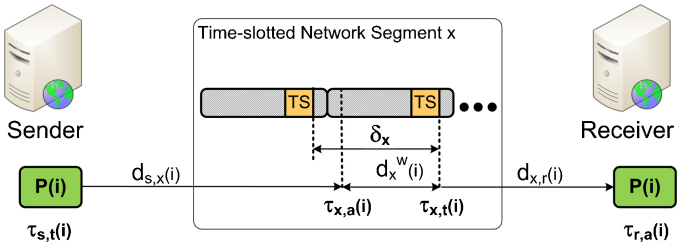

Fig. 1. One-way end-to-end delay (OWD) measurements having one time-slotted network segment in the measurement path

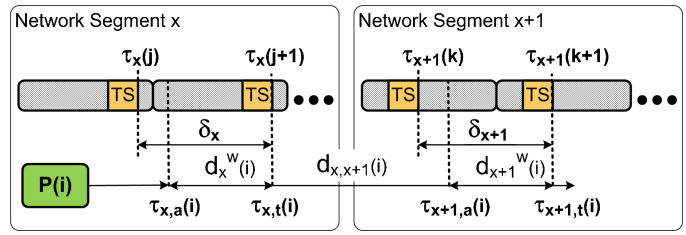

Fig. 2. Time-slotted randomness cancellation effect for successive time-slotted network segments.

network segment x, and therefore the wait time $d_x^w(i)$ for the next transmit timeslot $\tau_{x,t}(i)$, depends on the initial send time $\tau_{s,t}(i)$ and propagation delay $d_{s,x}(i)$. Send time $\tau_{s,t}(i)$ denotes network time, i.e., when the packet enters the network. However, a comparison of network timestamps acquired by tcpdump against application-reported timestamps for the setup presented in Section IV yields deviations of less than 60 μs for 99.98% of all samples. This is orders of magnitude below the probed randomness cancellation effect. Therefore, all delay diagrams in this paper's results section rely on application- or socket-layer acquired time stamps.

A first important observation from Fig. 1 concerns periodic stream packets. When using accurate sender clock and periodic streams with send period being a multiple of network period δ_x, the wait time $d_x^w(i)$ for all packets in a stream is almost constant. Moreover, when the variation of propagation delays $d_{s,x}(i)$ and $d_{x,r}(i)$ is low compared to the network segment period δ_x, the end-to-end delays will be almost identical. The end-to-end delay of all stream packets, therefore, depends primarily on the send time of the first packet within the stream.

Typical examples include ping ICMP echo requests, which by default transmit any full second, or voice over IP streams. However, the measured end-to-end delay with periodic streams is not representative for time-slotted networks, unless measurements are repeated while randomizing the first packet's start time.

In the following, this scenario is extended to the identical case of random streams and at least two time-slotted network segments. The fundamental problem presented in this paper is referenced in the following as time-slotted randomness cancellation effect. It originates from the fact that any measurement packet, sent at random start time, is time-synchronized with respect to absolute time modulo network segment period when leaving the first time-slotted network segment within its path. Propagation delays between subsequent network segments are commonly constant with small delay variations. Timeslots allocated in these network segments are synchronous to each other and to global time. This synchronicity results in almost constant wait times of the packet for the second and all subsequent network segments, which is why measurement packets fail to fulfill the IPPM start-time randomness requirements for these segments.

The network path of interest includes at least two time-slotted network segments x and x + 1, as shown in Fig. 2. Let the period of all time-slotted network segments in the path be an integral part of a second, synchronous with global time. The period duration should be significant when compared to the total end-to-end delay. This is certainly true for mobile cellular networks, where measurements illustrate 10 ms period for less than 20 ms minimum downlink (DL) OWD.

Let network segment x, left in Fig. 2, be the first time-slotted network segment in the packet path. Moreover, it can be assumed without loss of generality that the size of any packet P is less than the capacity of one timeslot TS.

Measurement packet P(i) arrives at network segment x at global time $\tau_{x,a}(i)$ and must wait for the next available timeslot for traversing network segment x. If the original packet send time satisfies the start-time randomness condition, the wait time $d_x^w(i)$ of packet P(i) for the next available timeslot will be a random value. The wait time fulfills the condition: $0 \leq d_x^w(i) < \delta_x$, where δ_x denotes network segment x's period.

However, when leaving network segment x at global time $\tau_{x,t}(i)$, packet P(i) is synchronous with all subsequent packets within this session relative to global time modulo δ_x. This is why, packet P(i) fails to fulfill the randomness criterion for network segment x + 1 and all subsequent network segments.

Due to global time synchronization of network segments x and x + 1, the wait time $d_{x+1}^w(i)$ of packet P(i) in network segment x + 1 depends on the propagation delay $d_{x,x+1}(i) = \tau_{x+1,a}(i) - \tau_{x,t}(i)$ between networks x and x + 1 and the constant relative time offset $\tau_{x+1,t}(i) - \tau_{x,t}(i)$ between timeslots in networks x and x + 1. All packets being synchronous with each other and with global time modulo δ_x, at starting point $\tau_{x,t}(i)$, a variation in the propagation delay $d_{x,x+1}(i)$ can increase the delay by multiples of network x + 1's period. In other words, $d_{x,x+1}(i) = d_{x,x+1}^{min}(i) + c*\delta_{x+1}$ with $d_{x,x+1}^{min}(i) = \min(\tau_{x+1,t}(i) - \tau_{x,t}(i))$ and c an integer value >= 0.

B. Round-Trip Delay in Mobile Cellular Networks

One particular application of this generalized problem is the measurement of RTD when both, request path and reply path include at least one time-slotted network segment. This is the case in mobile cellular networks that use periodical transmit time intervals (TTI) of 2 ms or 10 ms, depending on parameters like, e.g., the specific mobile modem hardware, throughput, or network state. Fig. 3 depicts this specific scenario. RTD measurement packet P(i) (e.g., a UDP or an ICMP echo request packet) sent by a mobile client at randomized start times arrives at random time $\tau_{UL,a}(i)$ at the time-slotted network segment and waits a random time $d_{UL}^w(i)$ for the next available time slot.

When leaving the network segment at time $\tau_{UL,t}(i)$, the packet is synchronous with all other packets within this session

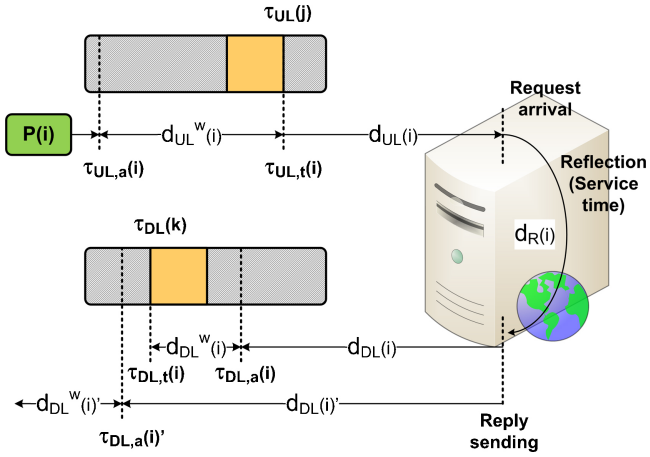

Fig. 3. RTD measurement packet synchronization in mobile cellular networks.

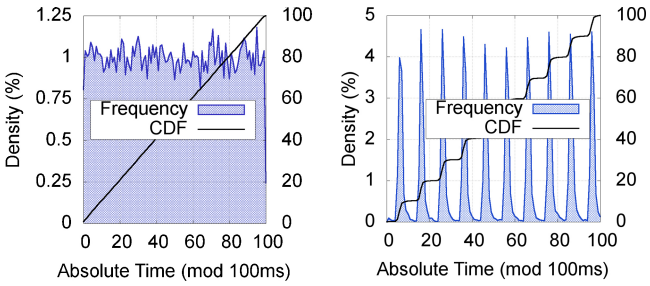

Fig. 4. Absolute time distributions (mod. 100ms) of uplink measurement samples at sending client and at receiving server (based on tcpdump network trace files). (a) Send time distribution (client). (b) Arrival time distribution (server).

and global time modulo uplink (UL) period δ_{UL}. The core network delay $d_{\mathrm{UL}}(\mathrm{i})$ to the reflecting server is low and the delay variation is small compared to the end-to-end delay and the period δ_{UL}. This effect, resulting in huge bias of time-slotted interfaces on the timing of measurement samples, can be confirmed by tracing network interfaces at the sending client and at the receiving server.

Using tcpdump-acquired global timestamps for a HSPA measurement session, Fig. 4 shows timestamp distribution and cumulative distribution (CDF) for random-start-time random-payload ICMP measurement samples modulo 100 ms.

The start-time distribution CDF curve in Fig. 4(a), captured at the network interface labeled as measurement point A in Fig. 7, confirms the expected, almost perfect uniform sampling. When capturing incoming samples at the measurement server (measurement point B in Fig. 7), the sampling is periodical, being heavily biased by the 10 ms network period as confirmed by the diagram in Fig. 4(b).

The statement about low delay variation holds true for the request packet's service (i.e., reflection) time $\mathrm{d_R(i)}$ within the measurement server and the reply packet's delay on its path back to the time-slotted link $\mathrm{d_{DL}(i)}$. Repeating the statement for the generalized case, request packets are global-time-synchronized by the UL, and therefore fail to meet the send-time randomness requirement for DL. Consequently, the DL

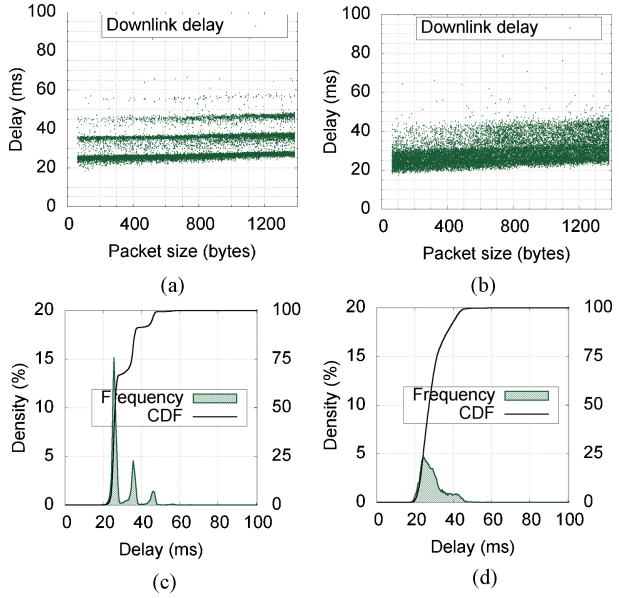

Fig. 5. Time-slotted randomness cancellation effect (left) versus true network behavior for RTD measurements in a mobile cellular HSPA network (DL). (a) Layered DL delay plot. (b) Expected "true" DL delay diagram. (c) Layered DL delay histogram. (d) "True" DL delay histogram.

delay extracted from a RTD measurement is expected to cluster around several values that differ by integer multiples of the DL period δ_{DL}. This is confirmed by Fig. 5, which uses results of random start-time random payload measurements, as presented later in Sections III-B, IV, and V, without randomness regeneration. The resulting DL delay extracted from RTD samples in Fig. 5(a) exhibits a layered structure, being seconded by histogram and CDF in Fig. 5(c) that confirm three main layers centered at 26 ms, 36 ms, and 46 ms.

Truly random measurement plots and histograms in Fig. 5(b) and (d), respectively, do not exhibit this layering. By true delay, we denote the delay response pattern that the specific link or network segment exhibits if tested in a hop-by-hop manner, safeguarding random start-time samples at the link's ingress. Measurement samples for Fig. 5(b) and (d) have been acquired using identical setup and methodology as referenced earlier for Fig. 5(a) and (c). Sole exception is the enabled server-based randomness regeneration support in the reflecting ICMP server as presented in Sections III and V. It is important to note that the presented effect does not depend on the measurement protocol but is intrinsic to time-slotted network operation, affecting any type of measurement or protocol.

C. Discussion

The findings presented in this section have severe consequences on delay measurements and estimations.

First and most severe, there is no solution to regenerate measurement randomness following a time-slotted network segment. Neither black-box nor white-box measurements, nor combinations of these two, commonly referred to as hybrid measurements, can regenerate randomness in intermediate nodes without adding explicit functionality to

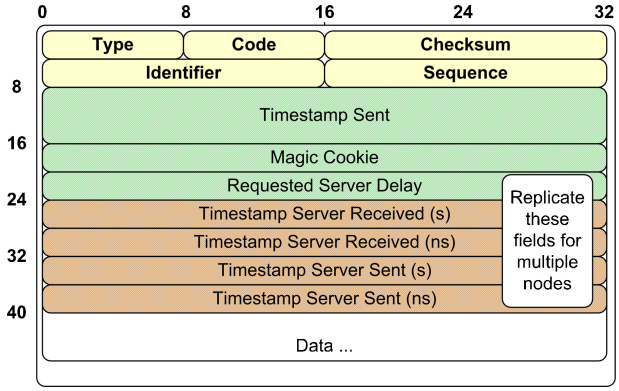

Fig. 6. Extended ICMPv4 header layout.

these nodes. The straight-forward solution to this problem, namely, repeated session establishment such that the networks allocate distinct time-slots and offsets, involves high costs in terms of measurement duration, complexity, and effort. This was the main trigger for developing the concept of randomness regeneration, which enables representative measurements, as detailed in the following sections.

Second, sequences of time-slotted networks can originate artificial network delay response for specific network segments. One-way delay in these segments does not exhibit continuous behavior but covers only a fraction of possible delay values. Results depend primarily on the specific session setup, the measured delay increasing in steps of integral network periods, as shown in Fig. 5(a). The origin of this layering is that a minimum variation in the server's reflection time or in intermediate network delay can increase the delay in subsequent segments by the duration of one period. This happens when a reply packet, because of this minimum delay variation, misses the timeslot k and must wait almost a full period δ_{DL} for timeslot $k+1$ to arrive. The corresponding scenario is shown at the bottom of Fig. 3, values marked as prime, $d_{DL}(i)$' and $d^{w}_{DL}(i)$'.

From a statistics point of view, averaging statistics like, e.g., mean or median delay, which rely on normal distributions, are not applicable or, at least, questionable from a methodological point of view for such multimodal distributions. Output of these statistics is misleading if not accompanied by a detailed description.

Finally, when estimating RTD using active (randomized) measurements in time-slotted networks, applications, and developers should consider that small variations in reflection (service) time can result in large discrete variations of the measured RTD value. Random-start-time ICMP echo RTD measurements can provide a satisfactory delay approximation for services with constant – almost zero – service time within the same session but fail to model the delay experienced by services with variable service times or for other sessions.

III. Ping++ Architecture

The proposed solution for accurate and representative delay measurements relies on insertion of artificially generated random delay within ingress nodes of time-slotted network segments. Main aim is to insert randomness into the measurement path, such that the measurement probes fulfill the start-time randomness requirements for any network segment. This solution cancels the global-time synchronization effect of time slotted network segments that has been described earlier and supports delay measurement samples to cover the entire representative range of possible end-to-end delay values.

A. Theoretical Model

Considering the network model, presented in Fig. 2, the proposed solution requires that any ingress node of a time-slotted network segment x must delay any incoming measurement packet P(i) by a random delay $d^{w,rand}_{x}(i)$. The random delay's value range equals the network segment's period: $d^{w,rand}_{x}(i) = rand[0..\delta_x)$. This approach allows accurate measurement of delay ranges even for periodic stream packets in time-slotted networks.

End-to-end measurement packet receivers are challenged to collect all artificially generated random delay values from the network path and subtract the sum from the effectively measured delay. In practice, this can be difficult or even impossible, as subsequent measurement packets might take distinct routes through the network.

An alternative, worthwhile approach is to store a set of client-proposed random delay values in the payload section of active measurement packets, such that compatible intermediate nodes can read and use these values. In addition, these active measurement packets can be used by intermediate nodes to store hop-by-hop timestamps, such that clients can determine exact hop-by-hop delays and delay variations. This solution has been adopted for the prototypical proof-of-concept implementation that is presented in the following subsection.

B. Practical Prototype Realization

As feasibility study for the proposed measurement methodology, a prototype has been designed and implemented that can assess accurate, representative RTD and OWD in mobile cellular networks. The active delay measurement prototype supports the Internet Control Message Protocol (ICMPv4 [2]), the User Datagram Protocol (UDP), and the Transport Control Protocol (TCP) as transport protocols for measurements. For UDP and TCP, it is required to run dedicated server applications in the network, which is why the remainder of this paper focuses on the ICMPv4 implementation.

The prototype extends standard ICMPv4 with respect to protocol, client, and server implementation. It is important to note that a mandatory prerequisite for accurate hop-by-hop delay measurements is that all intermediate nodes have accurate, high-resolution, global time synchronized clocks. The remainder of this subsection details on required changes.

ICMPv4 Protocol Modifications: Main challenge of the randomized delay insertion process is the collection of artificially generated random delay values from all ingress elements of intermediate network segments. To accomplish this task, the

prototype uses a minor nonstandard compliant extension of the ICMPv4 protocol. In particular, part of the ICMPv4 data field is used to store information for and from intermediate nodes. This modification conflicts with RFC 792 requirements on page 15: "the data received in the echo message must be returned in the echo reply message". However, this incompatibility was considered to be tolerable as it: 1) does not affect earlier implementations; 2) is used for a prototype demonstrating feasibility; and 3) must be standardized anyhow if deployed in the large.

Fig. 6 depicts the layout of the extended ICMPv4 header, all field values being stored in network byte order (NBO). The first 8 bytes, colored in yellow, match the standard ICMPv4 echo request/reply header. The header extension fields can be divided into sender-modifiable fields (colored in green, bytes 8–23) and fields that are supposed to be modified by intermediate nodes and/or the reflecting server (colored in red, bytes 24–39). The header is scalable in that bytes 20–39 can be replicated by the sending entity to accommodate for multiple time-slotted network segments within the measurement path. The implemented prototype uses only one extended header instance that is sufficient for accurate OWD and RTD measurements in mobile cellular networks.

The following list summarizes the ICMPv4 extension header fields:

1) *Timestamp sent:* A 64 bit unsigned timestamp, written by the sending ICMPv4 client, indicating the time since beginning of the epoch with nanoseconds resolution.
2) *Magic cookie:* A specific bit pattern that uniquely identifies the extended ICMPv4 protocol. Reflecting hosts artificially delay or write timestamps to an incoming ICMPv4 packet exclusively if the packet's bit pattern matches the server-configured magic flag. This feature ensures interoperability with standard ICMPv4 clients.
3) *Requested server delay:* A random value computed by the sending client, which instructs the ingress node or reflecting host to artificially delay the packet for the specific duration (in nanoseconds) before forwarding or reflecting it.
4) *Timestamp server received (s):* Timestamp when the server has received the packet. The value represents the offset in seconds relative to the value of *Timestamp sent*.
5) *Timestamp server received (ns):* Identical to previous field, nanoseconds fraction.
6) *Timestamp server sent (s):* Timestamp when the server has effectively forwarded the packet, in case, the effective wait time differs from the requested one. The value represents the offset in seconds relative to the value of *Timestamp sent*.
7) *Timestamp server sent (ns):* Identical to previous field, nanoseconds fraction.

For completeness, it should be mentioned that RFC 791 defines record route and timestamp features as part of the IP protocol. ICMP implementations can set the corresponding IP datagram option such that cooperating hosts can insert their addresses and/or timestamps into the IP header. However, the presented ICMPv4 extension header's complexity reflects that additional fields are required for improving measurement methodology.

ICMPv4 Client Modifications: Targeting accurate measurements and a flexible, portable implementation, the ping++ ICMPv4 client has been implemented from the scratch in C++ relying on the Boost [20] and Poco [21] libraries. In addition to ICMPv4, the new ping++ client supports UDP and TCP for message transfer. The new client implements a series of improvements with respect to the standard ping client, the most important being scenario-based message generation and support for ping flooding.

The so-called scenario, stored in a CSV file, defines a complete measurement session, including sequence of packets, packet send time, packet size, packet content, and artificial server delay for all packets sent within the session. Scenario files are commonly generated automatically by a newly implemented scenario generator, which computes earlier-mentioned fields randomly within preconfigured limits and/or according to specific distributions.

Generalizing, a scenario defines a specific representative traffic pattern of interest. Main benefits of the scenario concept are that: 1) measurements of typical protocol message flows can be implemented, matching closely an application's data exchange pattern and reflecting the application's behavior on a specific network, and 2) these use cases can be tested within subsequent measurements on their statistical relevance and repeatability. Real traffic can be emulated accurately by converting tcpdump or Wireshark traces into scenario files. However, care is required when creating scenarios based on capture files from intermediate network nodes. These traces could have been affected by randomness cancellation of prior time-slotted links, originating potentially undesired effects.

Ping flooding support refers to asynchronous message handling, i.e., decoupling of request sending from the client's receiving part. Ping++ clients generate asynchronous bursts of measurement samples without being required to wait for matching replies. This feature is a prerequisite for avoiding self-clocking effects and essential for measurements at high data rates.

ICMPv4 Server Modifications: besides the implementation of custom UDP and TCP servers to support RTD measurement using UDP and TCP, the Linux Kernel sources have been modified to implement the extended ICMPv4 functionality for any Linux system. Using the Sysctl interface, users can control extended ICMPv4 server functionality at runtime, e.g., control the server's read and write access to the extended protocol fields shown in Fig. 6. This functionality includes enabling or disabling artificial server delay functionality, controlling the writing of timestamps to ICMPv4 messages, as well as modification of the magic flag's value and various security settings, e.g., configuration of the maximum accepted server wait time.

C. Security Considerations

From a security point of view, the proposed solution undoubtedly introduces new potential threats, which, however, the authors consider to be manageable. On one hand, client-

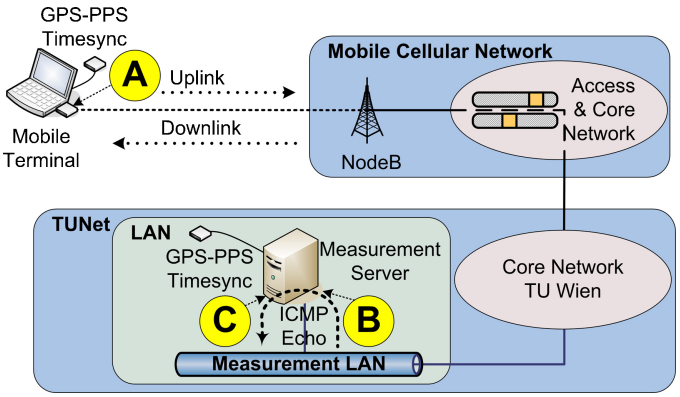

Fig. 7. Measurement setup RTD measurements in mobile cellular networks.

requested server delays result in allocation of server state and server load when compared to standard ICMPv4. Any kind of state establishes a new potential trapdoor for security and denial-of-service attacks.

On the other hand, the required random delay for eliminating time-slotted randomness cancellation is typically small, in the order of tens of milliseconds. Server-wait functionality in the Linux Kernel is disabled by default and features runtime-configurable maximum artificial server delay limits. This limit provides already a decent level of security. In case of high-risk or high-load network components, additional standard Linux security features can be easily deployed. The server could, e.g., cease to regenerate randomness and revert to standard ICMP echo or packet forwarding on detecting overload. More options include, but are not limited to, e.g., restrictions on total measurement traffic rate or limiting randomness regeneration service access to specific source IP addresses or address ranges.

In particular, when compared to security and privacy risks of alternative options that lack randomness regeneration but offer comparable functionality, e.g., packet capturing or traffic monitoring in intermediate nodes, the proposed solution offers substantial features at low additional security threat.

IV. Measurement Setup

Fig. 7 depicts the prototypical delay measurement setup. The public mobile cellular network of an Austrian telecom provider has been used as representative for typical time-slotted networks. The two measurement endpoints are standard PCs running Ubuntu Linux 12.04 with a custom compiled 3.4.0 Linux kernel at 1 kHz kernel tick rate and with PPS module support enabled. All measurements are static, i.e., the mobile terminal does not change its geographical position during measurements.

The server is a common of-the-shelf PC equipped with an ASUS P5B-VM (Intel I965G) mainboard, Intel CPU at 2.4 GHz, and 4 GB of RAM. As mobile client a Dell Latitude D630 laptop has been used, featuring an Intel CPU at 1.8 GHz and 2 Gbyte of RAM, connected to the mobile network using a Huawei E870 HSPA ExpressCard 34.

Mobile client and server are both accurately time-synchronized using EM-406A GPS modules with PPS support. The RS232 level conversion required for this GPS module is implemented by a SparkFun 8334 GPS evaluation board which has been extended by the PPS circuit as proposed in [8]. The Network Time Protocol Daemon (ntp) synchronizes both endpoints accurately with global time using the evaluation board's level-converted PPS and GPS time signals accessed by the kernel using native RS232 serial interfaces. Tests over several weeks have shown that this setup can synchronize the system clock accurately down to 5 μs to global time. Even if deployed in nonoptimum conditions (indoor, 1st floor, street view with 6-floor opposite building), the clock accuracy is better than 50 μs. This minimum limit for client and server time synchronization, as well as a minimum of five satellites in view during measurements is guaranteed by cross-checking with graphically processed NTP clocksync and loopstats log files.

Measurement points A, B, and C in Fig. 7 depict the specific interfaces within the measurement setup, where network-level tcpdump traces for Figs. 4 and 10 data have been acquired.

V. Measurement Results

Measurement results for fixed and mobile network technologies, including ADSL, HSPA, and LTE confirm the effectiveness of the proposed solution. The randomness regeneration methodology is demonstrated in the following using measurement results from a public HSPA network.

All measurement results in this paper base on the same measurement scenario file consisting of 20 000 random ICMP echo RTD packets. As detailed in Section III-B on ICMPv4 client modifications, the scenario file uses two independent random variables, such that ICMP packets of random payload size are sent out at random start times. Initial interpacket delay for this scenario is uniform distributed between 100 ms and 1000 ms whereas packet size is uniform distributed between 64 and 1400 bytes.

Measurement results are presented graphically either as x–y scatter plots, one point in the diagram representing one measurement sample, or as histogram function showing density superposed by the CDF. All diagrams are color-coded: UL results are colored blue, whereas DL results are green.

Fig. 8 depicts UL and DL delay extracted from state-of-the-art random-start-time ICMP RTD measurements as x–y scatter plot, showing delay as a function of payload size. The diagrams show a sharp DL delay layering in Fig. 8(b) and (d) as consequence of the methodological drawbacks presented in Section II-B. Identical drawbacks apply if the measurement setup is reversed, i.e., DL is measured first by starting the ping++ application on a network-based computer toward a mobile terminal that acts as reflecting server. In this case, the UL is the second time-slotted network segment and the measurement samples, global-time-synchronized by DL, fail to capture the real UL behavior as shown in Fig. 9(a) and (c).

Without server-based randomness regeneration, ICMP servers have almost constant reflection time as explained in Section II-B. Therefore, the distribution of measurement

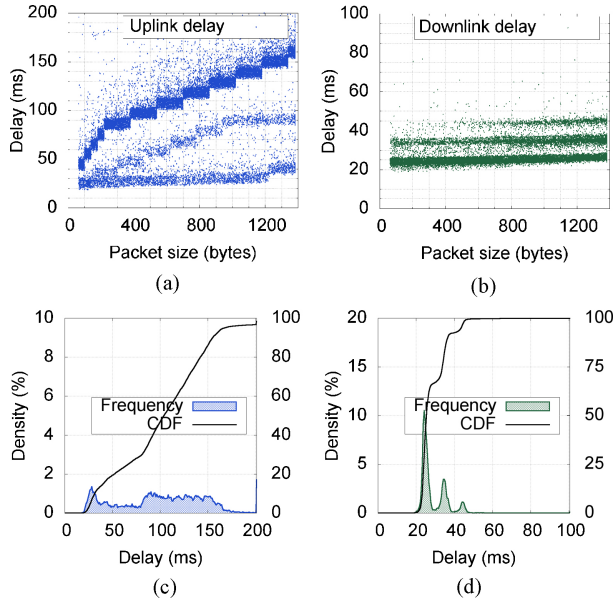

Fig. 8. Uplink and downlink delay for 20,000 random-start-time random-payload-size ICMP echo RTD measurements in a public HSPA network, (a) UL delay x-y scatter plot, (b) Layered downlink delay plot, (c) UL delay histogram and CDF, (d) DL delay histogram and CDF.

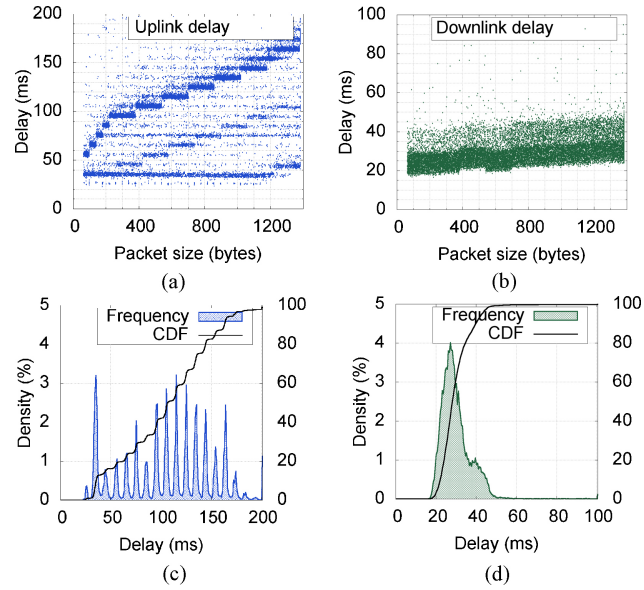

Fig. 9. Uplink and downlink delay plots and distribution functions for reverse measurement setup. (a) "Layered" UL (reverse setup). (b) "True" DL (reverse setup). (c) "Layered" UL histogram and CDF. (d) "True" DL histogram and CDF.

samples with respect to global time when leaving the reflecting ICMP server at measurement point C in Fig. 7 is identical to the one when entering the ICMP server. The corresponding diagram has been presented in Fig. 4(b).

Figs. 10 and 11 depict measurement results after deploying and activating the ICMP server extensions presented in Section III-B. The ping++ client sends ICMP echo requests to the ICMP server, proposing uniform distributed wait times between 0 and 10 ms according to the information stored in the scenario file.

A CDF comparison of ICMP server input sampling shown in Fig. 4(b) and server output sampling in Fig. 10(a) illustrates that server-based randomness regeneration does, indeed, substantially improve the start-time randomness timing of measurement samples for reply messages. Even if the zoomed representation modulo 10 ms, shown in Fig. 10(b) indicates existence of a density peak at 6 ms absolute time, it is expected that fine-tuning of client-proposed random wait-time values and better timer resolution should improve the quality of samples, i.e., their start-time randomness, even more.

Supported by the ICMP server's start-time randomness regeneration functionality, the corresponding OWD results shown in Fig. 11(a) for UL and Fig. 11(b) for DL represent the true end-to-end delay for the observed networks. Most important, the DL delay shape matches the shape of DL delay in Fig. 9(b) when DL was measured as first link with truly random start-time samples.

Although not relevant for the randomness cancellation effect, an important observation from a measurement methodology point of view is that HSPA UL is highly reactive in its nature. The UL delay in all presented diagrams, e.g., in Fig. 11(a), is layered due to flow state maintained by HSPA at layers below IP. Particular reason is that because of global network capacity optimization reasons, HSPA UL delay depends to a large extent on the user's recent load history and his current data transfer request. The higher the user-generated momentary load and recent load history, the higher is the UL capacity allocated by HSPA to the user and the lower the UL delay he experiences.

The highly variable data rate of random start-time random-payload measurement traffic triggers frequent HSPA UL state transitions in the presented UL diagrams. For instance, on arrival of several large-payload UL messages within a short time, the HSPA scheduler allocates additional capacity to the user. This higher capacity lowers the delay for subsequent messages, such that these will show up in one of the lower, slightly rising delay lines. Following short periods of inactivity or low traffic, the HSPA scheduler withdraws the grants, reducing the user's allocated capacity and increasing the delay of subsequent samples that will be positioned in the diagrams' upper delay lines. Detailed considerations on this topic can be found in previous publications [9], [10].

Moreover, comparing UL delay diagrams in Figs. 8(a) and 11(a) reveals that HSPA UL responses still differ significantly, despite using identical setup and identical measurement traffic over more than 3 h and 20 000 samples. An additional square step can be observed in Fig. 11(a) in the highest-delay pattern that starts close to the point ($x = 200$ bytes, $y = 80$ ms), causing a payload-delay offset for the following and all subsequent delay blocks.

The conclusion with respect to HSPA UL scheduling is that HSPA UL responds differently to identical user stimulus (i.e., data traffic) because of user-external factors like, e.g., time of day, cell load, and operator configuration. Additional measurements confirm earlier results [10] and the finding of other authors [11] and [15] that specific traffic properties such as constant bit rate or increased data rate improve the

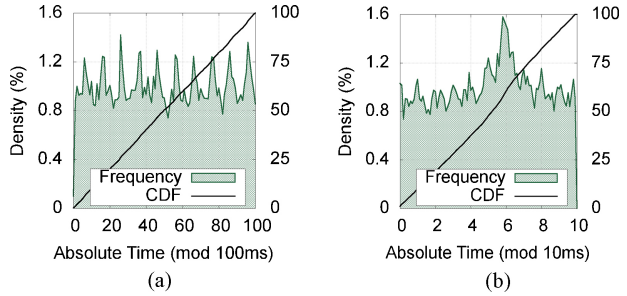

Fig. 10. Absolute time distributions (mod. 100 ms and mod 10 ms) of downlink measurement samples after server-based randomness regeneration (based on tcpdump network trace files, measurement point C in Fig. 7). (a) Time distribution. (b) Time distribution (zoomed-in view).

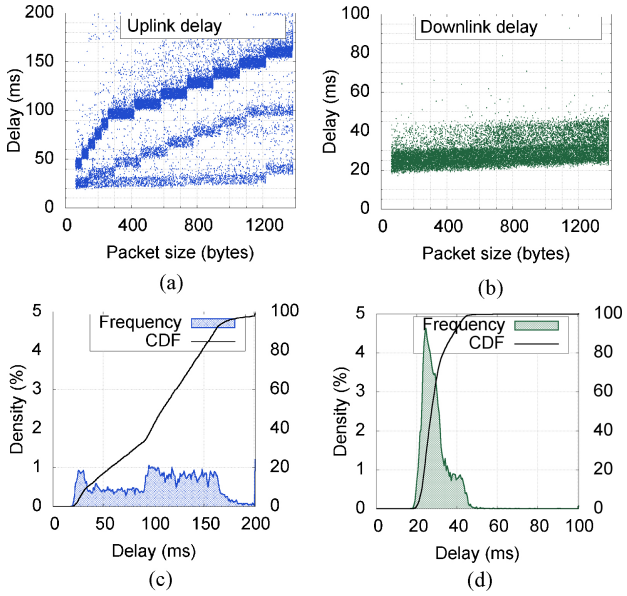

Fig. 11. Uplink and downlink delay measurement results when deploying random-delay server extensions on ICMP server and using the ping++ client. (a) "True" UL delay plot (ping++). (b) "True" DL delay plot (ping++). (c) "True" UL histogram and CDF. (d) "True" DL histogram and CDF.

repeatability and quality of delay measurements. Appropriate scenario definitions, e.g., packet send times and packet sizes computed to maintain a constant bit rate, can support to obtain unimodal delay patterns for HSPA UL.

However, it is questionable whether typical applications can shape their traffic to meet these demands, particularly if other applications on the same device generate traffic, too. Therefore, main finding with respect to representative delay measurement in networks featuring on-demand capacity allocation like, e.g., HSPA, is that measurements should optimally use the specific application's application's data stream – if available and definable – to produce an adequate network response.

VI. Conclusion

This paper confirmed that randomness - in particular random start time - is an indispensable prerequisite for state-of-the art delay measurements. Random start time measurements can detect huge systematic delay variation in time-slotted network links, which passes unobserved in nonrandomized measurements using periodical streams.

Main contribution of this paper is the finding that start-time-randomness is an essential but not sufficient requirement for representative delay measurements in sequences of time-slotted network segments. The reason behind is that the first time-slotted segment in the measurement path synchronizes random-start-time measurement samples such that a periodic sampled stream leaves the segment's egress. This heavily biased, periodic stream is unable to capture the representative behavior of subsequent time-slotted network paths.

Measurement representativeness can be increased by re-generating start-time randomness of measurement packets in ingress nodes of time-slotted networks. Enhancing traditional white-box or black-box-measurements, this methodology is required to capture the full, representative delay range of network paths under observation with a relatively low measurement effort. From a statistics point of view, a huge benefit of randomness regeneration is that the resulting distribution of delay samples avoids artificial multimodal layering.

Measurement results using the proposed methodology in mobile cellular networks confirm these findings. By injecting artificial random delay in the reflecting ICMP server, the tool can infer from RTD measurements onto the full delay range of RTD and hop-by-hop, i.e., UL and DL OWDs. The presented methodology does not depend on ICMP as measurement protocol but is specific to operation of time-slotted networks.

Concluding, appropriate methodology selection depends to a large extent on the point of view as pointed out by recent and ongoing IETF work [7]. Accurate delay measurements for networks that use on-demand capacity allocation require that measurement data matches the effective data stream for which delay is to be estimated or forecasted. If users are interested in representative delays, which applications will experience during ongoing and future sessions, they are advised to use randomness regeneration mechanisms in ingress nodes of time-slotted networks. Alternately, if measurement focus is on the delay behavior of periodic streams for the ongoing session, then start-time of the stream (i.e., the stream's first packet) should be randomized in order to detect possible interaction with network timing.

As future work the authors will approach the IETF IPPM Working Group to present and discuss the paper's main results. The requirement to extend intermediate nodes with dedicated measurement functionality in support of active and passive delay measurements can be at the origin of novel hybrid measurement architectures and protocols, which altogether improve delay measurement sample quality and representativeness.

Acknowledgment

The authors would like to thank the anonymous reviewers and gratefully acknowledge their valuable feedback, which has significantly improved quality and focus of this paper. The authors thank their colleagues Markus Laner and Philipp Svoboda for their support on PPS-based time synchronization

and many fruitful discussions. The authors also thank A1 Telekom Austria AG for their support by providing mobile hardware and SIM cards for the HSPA measurements.

REFERENCES

[1] L. De Vito, S. Rapuano, and L. Tomaciello, "One-way delay measurement: State of the art," *IEEE Trans. Instrum. Meas.*, vol. 57, no. 12, pp. 2742–2750, Dec. 2008.

[2] J. Postel. (1981, Sep.). *IETF Standards Track RFC 792: Internet Control Message Protocol* [Online]. Available: http://www.ietf.org/rfc/rfc792.txt

[3] V. Paxson, G. Almes, J. Mahdavi, and M. Mathis. (1998, May). *IETF Standards Track RFC 2330: Framework for IP Performance Metrics* [Online]. Available: http://www.ietf.org/rfc/rfc2330.txt

[4] G. Almes, S. Kalidindi, and M. Zekauskas. (1999, Sep.). *IETF Standards Track RFC 2679: A One-Way Delay Metric for IPPM* [Online]. Available: http://www.ietf.org/rfc/rfc2679.txt

[5] G. Almes, S. Kalidindi, and M. Zekauskas. (1999, Sep.). *IETF Standards Track RFC 2681: A Round-Trip Delay Metric for IPPM* [Online]. Available: http://www.ietf.org/rfc/rfc2681.txt

[6] C. Demichelis and P. Chimento. (2002, Nov.). *IETF Standards Track RFC 3393: IP Packet Delay Variation Metric for IPPM* [Online]. Available: http://www.ietf.org/rfc/rfc2681.txt

[7] A. Morton, G. Ramachandran, and G. Maguluri. (2012, Aug.). *IETF Standards Track RFC 6073: Reporting IP Network Performance Metrics: Different Points of View* [Online]. Available: http://www.ietf.org/rfc/rfc6073.txt

[8] M. Laner, S. Caban, M. Svoboda, and M. Rupp, "Time synchronization performance of desktop computers," in *Proc. IEEE Int. Symp. Precision Clock Synchronization Meas. Contr. Commun.*, Sep. 2011, pp. 75–80.

[9] J. Fabini, L. Wallentin, and P. Reichl, "The importance of being really random: Methodological aspects of IP-layer 2G and 3G network delay assessment," in *Proc. IEEE Int. Conf. Commun.*, Jun. 2009, pp. 1–6.

[10] J. Fabini, W. Karner, L. Wallentin, and T. Baumgartner, "The illusion of being deterministic: Application-level considerations on delay in 3G HSPA networks," in *Networking*, vol. 5550. Berlin, Germany: Springer, May 2009, pp. 301–312.

[11] M. Laner, P. Svoboda, P. Romirer-Maierhofer, N. Nikaein, F. Ricciato, and M. Rupp, "A comparison between one-way delays in operating HSPA and LTE networks," in *Proc. 8th Int. Workshop Wireless Netw. Meas.*, May 2012, pp. 286–292.

[12] M. Laner, P. Svoboda, E. Hasenleithner, and M. Rupp, "Dissecting 3G uplink delay by measuring in an operational HSPA network," in *Proc. Passive Active Meas. Conf.*, 2011, pp. 52–61.

[13] A. Barbuzzi, F. Ricciato, and G. Boggia, "Discovering parameter setting in 3G networks via active measurements," *IEEE Commun. Lett.*, vol. 12, no. 10, pp. 730–732, Oct. 2008.

[14] P. Romirer-Maierhofer, F. Ricciato, and A. Coluccia, "Explorative analysis of one-way delays in a mobile 3G network," in *Proc. 16th IEEE Workshop Local Metropolitan Area Netw.*, Sep. 2008, pp. 73–78.

[15] P. Arlos and M. Fiedler, "Influence of the packet size on the one-way delay on the down-link in 3G networks," in *Proc. 5th IEEE Int. Symp. Wireless Pervasive Comput.*, May 2010, pp. 573–578.

[16] H. Zheng, G. Rittenhouse, and M. Recchione, "The performance of voice over IP over 3G downlink shared packet channels under different delay budgets," in *Proc. IEEE 58th Vehicular Technol. Conf.*, vol. 4. Oct. 2003, pp. 2501–2505.

[17] D. Viamonte, A. Calveras, J. Paradells, and C. Gomez, "Requirements for the optimization of session setup delay and service interactivity in streaming services over 3G," in *Proc. 12th Eur. Wireless Conf. Enabling Technol. Wireless Multimedia Commun.*, Apr. 2006, pp. 1–10.

[18] S. Shalunov, B. Teitelbaum, A. Karp, J. Boote, and M. Zekauskas. (2006, Sep.). IETF standards track RFC 4656: A One-way Active Measurement Protocol (OWAMP) [Online]. Available: http://www.ietf.org/rfc/rfc4656.txt

[19] K. Hedayat, R. Krzanowski, A. Morton, K. Yum, and J. Babiarz. (2008, Oct.). IETF standards track RFC 5357: A Two-way Active Measurement Protocol (TWAMP) [Online]. Available: http://www.ietf.org/rfc/rfc5357.txt

[20] Boost C++ libraries [Online]. Available: http://www.boost.org

[21] Poco C++ libraries [Online]. Available: http://pocoproject.org

Joachim Fabini received the Diploma degree (Dipl.-Ing.) in technical computer sciences and the Ph.D. degree (Dr. techn.) in electrical engineering from the Vienna University of Technology, Vienna, Austria, in 1997 and 2008, respectively.

After five years of research and development in the telecom industry, he joined the Institute of Telecommunications (ITC) at the Vienna University of Technology in 2003 and is currently lecturing master level students and leading applied research projects. His current research interests include measurement methodologies and metrics in packet-switched wireless and wired networks, location-based services, and network architectures with a particular focus on machine-type communications, the session initiation protocol (SIP), and the IP multimedia subsystem (IMS).

Michael Abmayer received the Diploma degree (Dipl.-Ing.) in business engineering and computer science from the Vienna University of Technology, Vienna, Austria, in 2011.

He has been in the IT Consulting Business for more than ten years.