M2M Communication Delay Challenges: Application and Measurement Perspectives

Joachim Fabini* and Tanja Zseby
Institute of Telecommunications, Vienna University of Technology
Gusshausstr. 25/E389, 1040 Vienna, Austria
*Corresponding author: Joachim.Fabini@tuwien.ac.at

***Abstract*—During the last decades, networks have evolved from stateless copper wires to complex, stateful communication paths. Furthermore, the delay in wired and wireless access networks has decreased substantially, such that previously negligible delay contributions become relevant parts of the total end-to-end delay. This can affect critical M2M applications such as Smart Grid communication that depend on timely data transfer. Their demand for deterministic communication paths conflicts with the common, cost-driven trend to share and optimize *overall* network capacity.**

This paper assesses contributions and bias of network technologies and systems onto end-to-end delay in today's access networks. The delay measurement results that are presented for various access networks provide (a) methodological considerations for accurate one-way delay measurements in these networks, (b) delay estimates for specific technologies as a guideline for M2M communication network selection, and (c) exemplary technology-specific anomalies that M2M applications must be prepared to handle. Integrating these findings into M2M applications is key to improve their communication's robustness, performance, and ability to handle anomalies appropriately.

I. Introduction

Machine to Machine (M2M) communications denote automated exchange of information which involves at least two devices as communicating parties. This definition allows for a broad range of device types and communication patterns. The applications requirements with respect to communication networks may also differ substantially in terms of one-way delay, transfer capacity, delay variation, reliability, etc. Examples of M2M communications span from sporadic single-bit sensor readings to high-bit-rate video transfers and may include the need for elastic modifications.

On one hand, novel access network technologies beneficially support M2M communications in lowering one-way delay and increasing available network capacities. On the other hand the cost factor forces access network development and operation to focus on *overall* capacity optimization rather than on deterministic paths. There is a strong incentive to share existing capacities and dynamically allocate them based on local and global parameters.

Additional factors that complicate delay estimation and measurements in today's network include, but are not limited to subpath aggregation (I-1), network data optimization (I-2), and asymmetric link structure (I-3).

1) Subpath aggregation: When packets traverse aggregated access links, the scheduling strategies of these links influence each other. This is why the resulting delay can differ from the sum of isolated hop-by-hop delays. For instance a concentrator can aggregate data from several sensors organized in a mesh network and then forward the aggregated data to a centralized processing instance using cellular 3G links as a backhaul. In such aggregated subpaths, impairments of one path can interfere with behaviors of subsequent paths, generating artificial, systematic delay effects, which are hard to detect from an external observer's position. Examples include time-slotted paths and paths that perform on-demand capacity allocation like, e.g., High-Speed Packet Access (HSPA).

2) Optimization techniques: Today's access networks implement various optimization techniques, including demand-driven allocation of network capacity, time-slotted operation, data compression, and packet-based processing. They maintain state and history at layers below the IP layer such that application-experienced performance depends on current and previous network traffic and even on the packet payload content. This can cause huge delay variations that are difficult to isolate when viewing paths as a black-box.

3) Asymmetric links: Due to the asymmetric communication demands of some applications (e.g. sensor data collection) access networks often show different characteristics on the uplink and downlink. In such scenarios it is important for applications to consider one-way metrics instead of round-trip characteristics. The effort associated with one-way delay measurements is typically higher than for round-trip-delay, necessitating clocks synchronization and access to systems in remote locations.

A. Related Work

The Internet Engineering Task Force's (IETF) IP Performance Metric (IPPM) Working Group has published basic guidelines for accurate delay measurements in [2] and [3]. A recent update [4] defines an advanced stream and sampling framework for IPPM, addressing evolution of access networks and its impact on methodologies. De Vito et al. [1] present basic considerations on one-way delay measurement methodologies with focus on time synchronization requirements. Performance analysis of mobile cellular networks can be found in [6], [7]. In earlier and recent work [11], [12], [15] we have introduced randomness cancellation effects that occur in time-slotted networks.

Some recent publications focus on M2M communications. [14] and [8] propose methods that detect reactive behavior of networks, [9] analyzes the large-scale effect of M2M traffic patterns in cellular networks. [10] dissects M2M delay and [13] proposes scheduler optimizations for M2M in Long Term Evolution (LTE) networks with focus on the wireless layer.

This publication contributes a detailed analysis on challenges of time-critical M2M communications and measurements in today's access networks. The analysis is complemented by sample setups and measurement guidelines to determine the impact and order of magnitude of systematic uncertainty factors in these networks.

B. *Structure of this Paper*

In the remainder of this paper, the concise summary of M2M communication requirements from application and measurement perspective in section II is followed by analysis and proposals on how M2M communications and measurements can master access network evolution in section III. Results in section IV are followed by the final conclusions in section V.

II. M2M Communication Requirements

This sections summarizes some requirements and challenges with respect to delay that measurements and M2M applications may face in access networks.

A. *Application Perspective*

The following requirements can play a significant role in time-critical M2M applications:

1) Low end-to-end delay: Many M2M applications perform better if their communication delay is low. Examples are critical real-time control loops in Smart Grid implementations.

2) Deterministic delay: Real-time applications can benefit from communication delay that has low variation (jitter) and is independent of current and preliminary network and system state. Negative examples include cellular networks like HSPA that implement on-demand capacity allocation. This strategy improves overall network performance at the cost of state-dependent delay from a single application's perspective.

3) Bounds on systematic delay variations: Time-critical M2M applications may require minimum and maximum systematic delay bounds for a specific path. These bounds are difficult to determine by means of measurements. Some access networks use periodic time-slotting or similar time-related scheduling operations with substantial service time (e.g., any 10 ms). These effects can cause systematic timing interference between subsequent links of aggregated paths [15].

4) Linear delay-payload dependence: Many algorithms rely on the linear delay equation *delay = packetSize/transferRate*. But today's networks have packet-based transfer characteristics that might differ from this linear behavior. M2M applications are advised to consider these peculiarities, which, in some cases, might help them to substantially lower their end-to-end delay. As will be shown in section V, applications can speed up data transfer by exploiting knowledge on their communication path characteristics.

B. *Measurement challenges*

Network measurements are essential for the operation of M2M communications. But technological advances of networks directly impact measurement methodologies.

1) Time: Correct M2M application operation may depend on absolute and relative time, i.e., on information about network delay and on timestamps of its peers. Asymmetric links and data traffic characteristics mandate one-way delay measurements and synchronized clocks in all systems But accuracy of timestamps and therefore delay depends on several factors. Besides timer resolution, hardware, and operating system, the Internet Engineering Task Force's (IETF) definition of host time and wire time leaves room for interpretation. In particular, host-based timestamps for measurements can be acquired either in application space, in kernel space, or within network drivers. For M2M applications that require low bounds on communication delay these distinct timestamp acquisition variants can be critical.

2) Aggregated paths: Superposition and reciprocal influence of timing and optimization effects in subsequent links of a path can impair packets and measurements. Main challenge is to counteract randomness cancellation [15], i.e., to safeguard that all links that form a communication path are measured using random start time samples. Ideally, hop-by-hop measurements should assess all links with random samples.

3) Representative delay: The IETF IP Performance Metrics (IPPM) Framework addresses common characteristics of metrics and methodologies like accuracy, repeatability, and continuity. M2M applications can significantly reduce their measurement effort if their communication paths behave deterministic. This includes, e.g., information that measurement results at a time *t* are valid at a later time, too, or that packet delay on a path increases linearly with payload.

III. Mastering Delay Challenges

Based on the challenges that access networks impose to M2M applications and measurements, this section analyzes potential uncertainty and impairment factors in detail. At its center is the following question: *Which are the main limiting factors for networked applications or active measurements that, even in the absence of competing traffic, network congestion or other impairments, can obscure real, transient network delay effects in access networks?* And, in addition: *can we quantify main systematic delay contributions and uncertainties in active measurements or at least determine their order of magnitude by choosing the right methodologies?*

1) Timestamps: RFC2330 [2] differentiates between *host time* and *wire time* when timestamping a packet. However, we argue that both definitions are subject to interpretation.

Fig. 1 depicts an end-to-end packet transfer between host A on the left and host B on the right, illustrating three common timestamping variants for **host time**. Applications acquire timestamps tA_A at application layer. Driver timestamps tD_A can use higher resolution timers, typically in kernel space, which is closer to the network in terms of delay, whereas tcpdump and PCAP traces timestamp packets tP_A right before they are passed to the network device. The difference between these three timestamps depends primarily on the host's hardware, operating system, configuration, and on timer resolutions available at various layers.

Applications, measurement protocols and their objectives decide on feasible timestamp acquisition methods. End-to-end

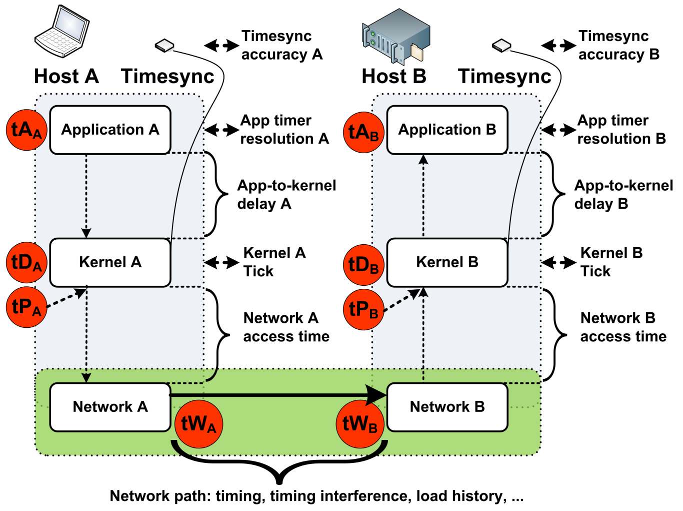


Fig. 1. Overview: Impairment factors and position of timestamp acquisition interfaces in M2M communications and end-to-end delay measurements

measurements will prefer time tA_A, whereas pure network delay is better reflected by tcpdump timestamp tP_A.

M2M communication protocol design limits possible timestamp options, too. As first option, receivers can compute one-way delay in real-time when applications or active measurements insert timestamps into packet headers or payloads. For this purpose, applications can insert the timestamp (tA_A) before sending a packet. Alternatively, dedicated kernel-level drivers can insert accurate timestamps at kernel level (tD_A) at the cost of additional implementation effort. These methods can interfere with security measures as detailed later on. As second option the sending host stores timestamps locally and correlates them later on with receiver timestamps. This less intrusive mechanism, which is used by passive measurements, is applicable when either immediate feedback is not mandatory, or when packets have no sender timestamp field. Any of the options (tA_A, tD_A, tP_A) is possible and may be chosen according to the measurement objectives.

The IETF concept of **wire time** – as first (or last) appearance of packet bits at an observation point – is ambiguous if used in wireless and software-dominated networks. First, the position of the observation point is not specified. Second, complex scheduling algorithms govern access to networks. Sent packets spend significant time in the sender's software drivers while waiting for the access scheduler to grant them access to the network. We argue that the driver waiting time is a network technology-dependent design parameter and therefore should be part of the network delay. The concept of wire time misses this delay, which can be substantial. As example time slotting in HSPA networks results in up to 10 ms wait time in the driver vs. 20 ms minimum downlink delay.

2) Clock Synchronization: One-way delay measurements according to [3] mandate synchronous clocks in all measurement endpoints. Low-cost solutions using Global Positioning System (GPS) Pulse per Second (PPS) signals can synchronize local computer clocks accurately down to 1-10 μs to global UTC time [7]. Own measurement results yield 10-50 μs accuracy to be achievable with low-cost devices in non-optimum GPS reception conditions. Expensive GPS receivers with roof antennas permit accuracies better than 1 μs. Alternatively, relative clock synchronization based on the IEEE 1588 Precision Time Protocol (PTP) can be used in low-latency networks like Ethernet.

3) Application to network delay: Application-to-kernel delay in fig. 1 maps to the difference $tD_A - tA_A$. It depends mainly on hardware and operating systems. Empirical evaluation on a Linux system (Intel Core2 Duo at 1.8 GHz) with variable payload packets (64-1400 byte payload) yields a difference tP_A - tA_A of 40 μs median and below 55 μs for 99.8% of all 20,000 measurement samples. Small payloads have about 10 μs lower application-to-kernel delay than larger payloads. For many M2M applications and measurements this delay should be tolerable. Advanced socket calls enable applications to acquire socket-layer timestamps which have been verified to be identical at microsecond precision to the ones reported by tcpdump. However, this methodology does not work if timestamps must be inserted into an outgoing packet, a case in which we recommend to consider custom kernel-level driver timestamping of measurement traffic as an option to eliminate application-to-kernel delay.

4) Randomness cancellation effects: Time-slotted network segments or components in the measurement path synchronize measurement packets with global time. When leaving such a time-slotted global-time synchronized network entity, measurement packets do no longer fulfill the IPPM's start-time randomness condition. As main consequence probing based on random sampling is not possible beyond the first time-slotted segment of a network path. Measurement packets therefore can no longer assess the full delay range of subsequent network segments. This limits measurement sample representativity to the specific measurement session. As solution we propose in [15] repeated session establishments or active measurement support by randomness re-generation in intermediate nodes.

5) Reactive network behavior: Mobile cellular network operators optimize their networks primarily on overall capacity and throughput, i.e., revenue. Wireless spectrum is limited and must be used economically. Therefore HSPA schedulers allocate wireless capacity on user request and de-allocate it within fractions of a second. User experience of a single mobile user or an application therefore depends on a series of local and global uncertainty factors like user-generated load history, users in the cell, cell load, etc. In particular sporadic packets might suffer from higher delay in reactive networks. Reasons include the use of low-capacity signaling channels for transferring small packets or the time needed to re-allocate channel capacities. HSPA scheduler policies favor greedy users: the more data a device transfers, the more resources the scheduler grants to it and the lower the user-experienced end-to-end delay is. The delay difference can be up to tens or hundreds of milliseconds, higher data rates being an option for increasing determinism [11].

6) Network Optimizers: Compression algorithms can hinder reliable and representative network performance measurements. To increase overall network capacity, some mobile operators compress traffic on wireless cellular links by deploying either server-only or client-server optimizers. These software modules use compression techniques to reduce the effective packet size. Although compression focuses on specific high-traffic, high-volume protocols like HTTP and FTP, or file types like JPG images, it can theoretically compress measurement packets, too. This can lead to delay variations depending on the specific packet payload. Using a highly compressed, random measurement packet payload (e.g., a ZIP archive segment) ensures that measurements capture real network performance and are not biased by network optimizers.

7) Security Challenges: Security mechanisms like IPsec, TLS or SSL challenge accurate timestamps. Encryption at application or transport layer introduce additional delay and prevent both, kernel-level drivers to insert timestamps into packets, and tcpdump to identify packets for passive timestamping.

8) Limitations Summary: Table I summarizes challenges and uncertainty factors in accurate and representative measurements and gives an order of magnitude of the introduced uncertainty to be expected in today's systems according to our empirical measurement results.

IV. Measurement Results

A. Measurement Setup

Measurement results are acquired using the Representative Delay Measurement (RDM) Tool [15]. RDM is a C++ client-server application that generates measurement packets according to a pre-computed scenario, which defines send time, payload size, and other parameters. For the tests below RDM uses ICMP as transmission protocol. The reflecting ICMP server includes a kernel patch to prevent randomness cancellation. On this purpose it artificially delays incoming packets for a packet specific, client-proposed interval before reflecting them. Client and server run Ubuntu 12.04 using Linux kernel 3.13 with 1khz system tick. For VDSL and UMTS measurements the RDM server was connected to the public Internet using a Gigabit Ethernet interface ending in the Vienna University of Technology backbone. The Technicolor TG788vn VDSL modem was restricted to 768 kbit/s uplink and 8 Mbit/s downlink capacity. Clients and servers synchronize to global time using GPS-PPS receivers.

The pre-computed measurement stream consists of 20,000 packets with inter-packet intervals of 100-1000 ms, payload size of 64-1400 bytes, and artificial server wait time of 0-9999 μs. Total scenario duration is 11032 seconds and average scenario data rate is 10.61 kbit/s. Randomness in terms of inter-packet delay and payload size lets transient network effects spread over the entire payload range instead of affecting a limited payload region. x-y scatter plots display one-way delay results as a function of payload size. One point in the diagrams corresponds to one measured delay value. Blue color represents uplink and green color downlink delay results.

B. Ethernet Delay

A crossed patch cable that connects two on-board Ethernet interfaces has sufficiently low delay to illustrate limiting factors in active software-based delay measurements. Time-critical M2M applications will experience such effects, too, namely: bias by time synchronization, application-to-kernel delay, and randomness cancellation.

First delay measurements have reported an asymmetry in forward and reverse Ethernet delay. Uplink delay exceeds downlink delay by almost 50 μs when comparing fig. 3(a) and 3(b). Under the premise of symmetric Ethernet link behavior, analysis of RDM's implementation and of the measurement setup identified two possible sources of the asymmetry: either (a) time synchronization, or (b) application-to-kernel delay.

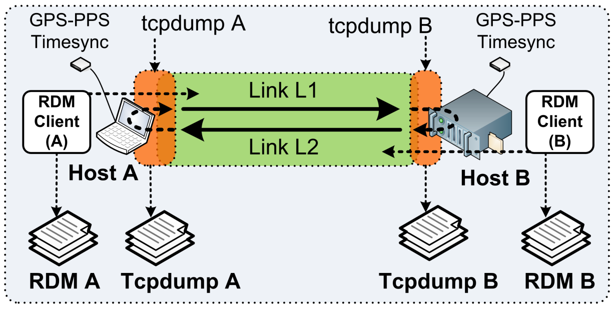


Fig. 2. Accuracy Test: Measurement of One-way and Application-to-Kernel Delay for Cross-connected Ethernet Patch Cable using two RDM clients and tcpdump.

A more complex measurement instrumentation is needed to identify the culprit. Two identical hosts A and B are connected using a crossed patch cable at 100 Mb/s, interfaces having configured IPv4 addresses as shown in fig. 2. The hosts have identical hardware and software setup, cloned package lists and custom Linux kernel, but separate GPS PPS time synchronization units. One RDM client runs on host A (denoted as RDM A) and one on host B (RDM B). They both generate identical traffic using the 20,000 sample scenario. Host A reflects traffic of RDM B, and host B the one of RDM A. Tcpdump instances on host A (tcpdump A) and on host B (tcpdump B) record all traces and accurate timestamps at network level. For the discussion we assume that the Ethernet cable is composed of two unidirectional logical links, L1 and L2. L1 transfers packets from host A to host B, and L2 the ones from host B to host A. Packets originated by RDM A measure L1 as forward link and L2 as reverse link, whereas RDM B traffic measures L2 as forward link and L1 as reverse link. Packet type (ICMP echo request or reply) has no bias on delay. RDM clients A and B report their respective forward and reverse link delays, complemented by tcpdump traces of hosts A and B.

Fig. 3 shows the measurement results reported by the RDM clients. The forward link delay reported by both RDM clients A and B (fig. 3(a) and 3(c)) is by more than 50 μs higher than their reverse link delay (fig. 3(b) and 3(d)). This symmetry identifies variant (b), application to kernel delay, as culprit. On time synchronization troubles – variant (a) – RDM A's forward link delay would equal RDM B's backward link delay and vice-versa. Finally, the delay derived from tcpdump traces on hosts A and B in fig. 4 confirms this finding. Effective socket-layer delays for link L1 and L2 are almost identical.

Analysis of the RDM client's source code revealed that the sending timestamp is inserted into the ICMP request's payload section *before* sending the packet, in application space (tA_A in fig. 1). The delay between the socket call in application space, the context switch to kernel space and the packet being handed over to the network driver in kernel space causes a 50-55μs increase of the packet's forward link delay. The other three timestamps per packet, including the reflecting server's receive and send timestamp and the client timestamp when receiving the packet are inserted in kernel space at locations tD_B, tD_B, and tD_A of fig. 1, respectively.

Challenge	Description, options	Consequences	Value	Solutions
Timestamping location	Location in host where the timestamp is added	Timestamp value and accuracy varies with timestamping location	20 μs–10ms	Chose timestamping location according to measurement objectives.
Clock synchronization	Clock synchronization is central to delay measurement, either relative (involved systems) or global to UTC.	Clock synchronization affects all timers and timestamps within a system.	1–50 μs for GPS-PPS	Good quality GPS-PPS receivers with clear view to the sky, adequate cable length compensation.
Measurement protocol	Timestamps inserted into packets or stored locally at the sender	Determines layer at which timestamps can be acquired	20–100 μs	tcpdump or kernel timestamps recommended for network delay.
Kernel tick	How often applications are triggered and have the opportunity to send data.	Depends on OS and configuration. Typically 100 Hz, 250 Hz,1 kHz.	1ms – 10ms	Configure high kernel rate (1 kHz) and turn tickless feature off.
Application to kernel delay	Delay to copy packets from user- to kernel space	Delay affects any packet	20–100 μs	Acquire timestamps in kernel space if feasible
Time-slotted network paths	Time-slotted networks offer periodic service opportunities, data can be sent periodically	Packet waits up to one network period for access. Random sampling impossible beyond first t.s. link	0–20ms, depending on technology.	Use hardware supporting low TTI (period), randomness re-generation in intermediate nodes [15].
Reactive networks	Network schedulers allocate user capacity on-demand	Sporadic packets have higher delay than high bitrate streams. Penalty up to hundreds of ms in HSPA.	Depends on network type and config.	Use high-bitrate streams. Reproduce exact application stream for accurate measurement results.
Network optimizers	Optimizers compress packet data (lossy or lossless), focus on common file formats like jpeg or html.	Depending on the payload type, measurement packets might be compressed silently on subpaths.	Depends on link capacity & opt. objectives	Avoid optimizers in the path, use non-compressible file types (encrypted ZIP archive file)
Asymmetric links	Many network technologies provide asymmetric links with downlink capacity exceeding uplink capacity.	Different packet delays on forward & reverse links. Forward link can bias sampling for reverse link.	Depends on network technology	Assess downlink and uplink using separate streams or do randomness regeneration in reflecting nodes.

TABLE I. SUMMARY OF CHALLENGES, IMPAIRMENT FACTORS AND SOLUTIONS IN M2M COMMUNICATIONS AND MEASUREMENTS

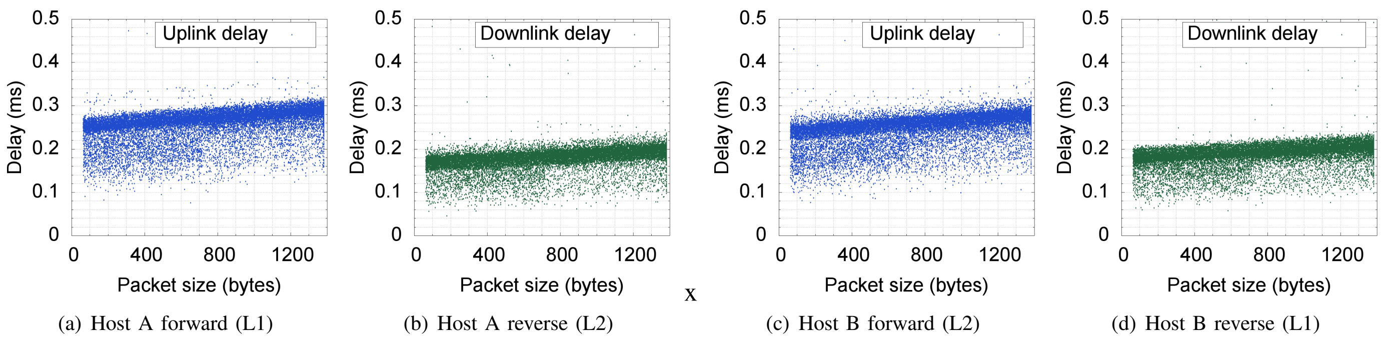


Fig. 3. One-way delay for crossed patch cable Ethernet (RDM-client reported): first timestamp acquired in application space, all others in kernel space.

Solutions include either the use of tcpdump instead of application-level measurements or the programming of custom kernel-level network drivers which insert the timestamp in kernel space into the measurement packet payload.

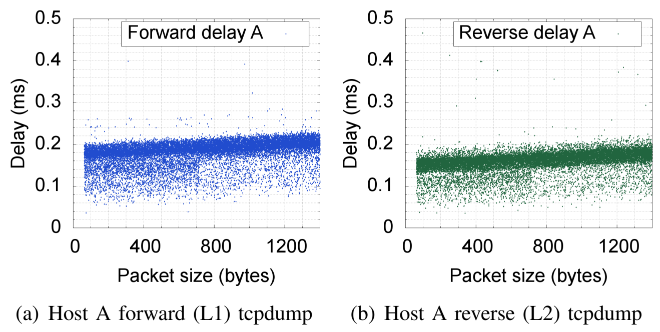


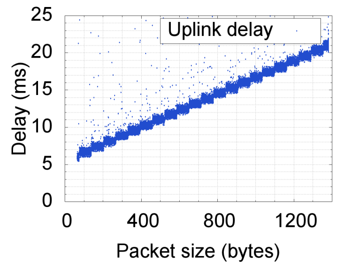
Fig. 4. One-way delay for crossed patch cable Ethernet (tcpdump-reported)

The following conclusions can be drawn from the Ethernet tests: (a) RDM delay accuracy equals tcpdump timestamp difference for the reverse link and is about 50μs higher for the forward link. (b) For application-level timestamping, application-to-kernel delay adds 100μs penalty per one-way end-to-end delay. (c) Applications can read kernel-assigned timestamps using advanced socket calls (ioctl, SIOCGSTAMP) to improve quality and accuracy. (d) Active measurement protocols should be designed such that network-driver-level insertion of timestamps into measurement packet payload is possible, considering potential introduction of security leaks by conflicting with application-level checksums and signatures. (e) Low-cost time GPS-PPS hardware clock synchronization provides good accuracy to UTC, well below 50μs.

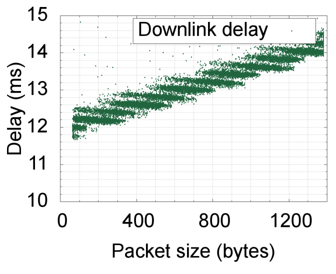
C. VDSL

The diagrams presented in fig. 5 show that VDSL has a clearly structured uplink and downlink delay shape, exhibiting a highly deterministic behavior. For clarity the two downlink delay plots in fig. 5(b) and 5(c) use a zoomed scale. In earlier measurements we have noticed that HSPA uplink delay decreases when increasing inter-packet delay. This matches the findings of [8]. Comparing fig. 5(a) (low-rate scenario) and 5(d) (high-rate scenario) shows that VDSL performs no reactive behavior at all for the tested measurement stream rates. Its one-way delay depends primarily on the packet payload. The steep increase of delay with increasing payload size is conditioned by the relatively low uplink capacity of 768 kbit/s.

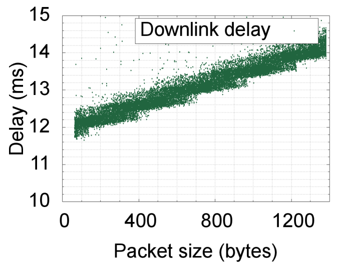
In [15] we have reported strongly biased HSPA measurement samples for downlink due to randomness cancellation effect in the uplink. Comparing VDSL downlink one-way delay diagrams with and without activated server randomness re-generation in figures 5(b) and 5(c) confirms that VDSL is affected by time-slotted randomness cancellation, too. However, the order of magnitude of the impairment is lower then



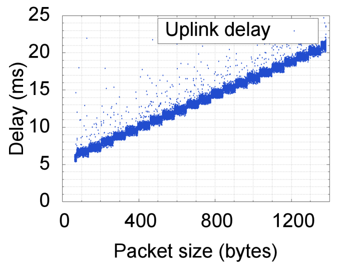
(a) VDSL Uplink Delay



(b) VDSL Downlink Delay



(c) VDSL DL Delay (Rand)



(d) VDSL UL (High Rate)

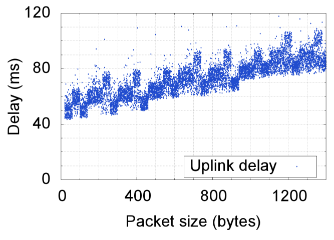
Fig. 5. One-way delay in VDSL networks.

for HSPA because of VDSL's lower network service time.

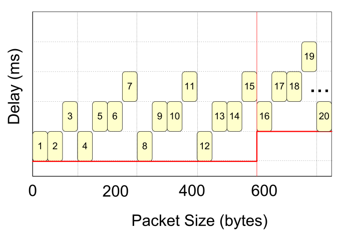
Worth noting is the difference of VDSL uplink and downlink delays in fig. 5(a) and fig. 5(b). We expected one-way delay ratios which map to the inverse of the link capacity ratio (768 kbit/s UL vs. 8 Mbit/s DL). Therefore it surprises that uplink delay for 64 byte packets amounts to half of the downlink delay. The reason for this apparently paradox behavior is active interleaving on the VDSL downlink whereas the uplink operates non-interleaved. As confirmation, a temporary activation of VDSL uplink interleaving has yielded an average uplink delay of 22.5 ms for 64 byte packets, which is about four times the uplink one-way delay without interleaving.

V. Conclusions

This paper shows that communication paths, hardware, operating system, applications of both, sender and receiver systems, are potential sources for systematic delay variations. M2M applications and delay measurement methodologies have to make their decision for a specific type of timestamp explicit and use it consistently to achieve repeatability. We recommend to use *network delay at socket layer*, i.e., the difference of tcpdump timestamps, as a representative, objective compromise for network delay in complex, software-dominated networks.



(a) UMTS 384 kbit/s Uplink



(b) UMTS 384 kbit/s UL Scheduling

Fig. 6. UMTS uplink 384 kbit/s dedicated channel scheduling.

Still, M2M applications and -measurements must be prepared to handle and can beneficially exploit anomalies in specific access networks to lower their effective end-to-end delay. A first anomaly has been presented in section IV-C. When interleaving is activated for VDSL it results in the delay to quadruplicate vs. the non-interleaved delay. As second example we refer to the payload-delay pattern of UMTS 384 kbit/s dedicated uplink channel in fig.6(a). The delay of 450 byte packets in this diagram is systematically by 20 ms *lower* than the delay of smaller 400 byte packets. This uncommon transfer curve is due to the UL scheduler that transfers 1, 2, 4, or 8 blocks within one transmit time interval (TTI) as illustrated by fig. 6(b). Packets which fit into 7 timeslots require 3 TTIs to transmit (1+2+4) blocks. M2M applications could exploit this anomaly and substantially lower their end-to-end delay by slightly *increasing* their packet payload. With evolving complexity of access networks we expect similar uncommon transfer patterns to be implemented more often in the future.

References

[1] L. De Vito, S. Rapuano, L. Tomaciello, *One-Way Delay Measurement: State of the Art*, "Instrumentation and Measurement, IEEE Transactions on", vol.57, no.12, pp. 2742–2750 (2008).

[2] V. Paxson, G. Almes, J. Mahdavi, M. Mathis, *Framework for IP Performance Metrics* Network Working Group RFC 2330, May 1998.

[3] G. Almes, S. Kalidindi, M. Zekauskas, *A One-way Delay Metric for IPPM* Network Working Group RFC 2679, Sept. 1999.

[4] J. Fabini and A. Morton, *Advanced Stream and Sampling Framework for IPPM* Network Working Group RFC 7312, Aug. 2014.

[5] M. Laner, P. Svoboda, E. . Hasenleithner, M. Rupp, *Dissecting 3G Uplink Delay by Measuring in an Operational HSPA Network*, "Proceedings of the Passive and Active Measurement Conference, PAM'11", Springer, Atlanta, GA, pp. 52–61 (2011).

[6] M. Laner, S. Caban, P. Svoboda, M. Rupp, *Time Synchronization Performance of Desktop Computers*, "Precision Clock Synchronization for Measurement Control and Communication (ISPCS), 2011 International IEEE Symposium on", pp. 75–80 (2011)

[7] M. Laner et al., *A Comparison Between One-way Delays in Operating HSPA and LTE Networks*, "Proceedings of the 8th International Workshop on Wireless Network Measurements WinMee'12", pp. 286–292 (2012)

[8] M. Laner, P. Svoboda, and M. Rupp, *A benchmark methodology for end-to-end delay of reactive mobile networks*, Wireless Days (WD), 2013 IFIP, pp. 1–8 (2013).

[9] M. Z. Shafiq, L. Ji, A. X. Liu, J. Pang, and J. Wang, *Large-scale Measurement and Characterization of Cellular Machine-to-machine Traffic*, IEEE/ACM Trans. Netw., vol. 21, no. 6, pp. 1960–1973 (2013).

[10] N. Nikaein and S. Krea, *Latency for Real-Time Machine-to-Machine Communication in LTE-Based System Architecture*, in Wireless Conference 2011 - Sustainable Wireless Technologies (European Wireless), 11th European, pp. 1–6 (2011).

[11] J. Fabini, W. Karner, L. Wallentin, T. Baumgartner, *The Illusion of Being Deterministic - Application-Level Considerations on Delay in 3G HSPA Networks*, "Networking 2009", Springer, Lecture Notes in Computer Sciences, Volume 5550/2009, pp. 301–312 (2009)

[12] J. Fabini, L. Wallentin, P. Reichl, *The Importance of Being Really Random: Methodological Aspects of IP-Layer 2G and 3G Network Delay Assessment*, "Communications 2009", ICC '09. IEEE International Conference on, pp. 1–6 (2009)

[13] A. Gotsis, A. Lioumpas, and A. Alexiou, *M2M Scheduling over LTE: Challenges and New Perspectives*, IEEE Vehicular Technology Magazine, vol. 7, no. 3, pp. 34-39 (2012).

[14] P. Svoboda, M. Laner, J. Fabini, M. Rupp, F. Ricciato, *Packet Delay Measurements in Reactive IP Networks*, IEEE Instrumentation & Measurement Magazine, vol.15, no. 6, pp. 36–43 (2012).

[15] J. Fabini, M. Abmayer, *Delay Measurement Methodology Revisited: Time-slotted Randomness Cancellation*, Instrumentation and Measurement, IEEE Transactions on, vol.62, no.10, pp. 2839–2848 (2013).