On Limits of Embedded Systems in Network Packet Processing

Ondrej VONDROUS, Peter MACEJKO, Zbynek KOCUR

Department of Telecommunications, Faculty of Electrical Engineering, Czech Technical University in Prague, Technicka 2, 106 27 Prague, Czech Republic

ondrej.vondrous@fel.cvut.cz, peter.macejko@fel.cvut.cz, zbynek.kocur@fel.cvut.cz

DOI: 10.15598/aeee.v14i4.1835

Abstract. *The paper deals with a measurement of single-hop one way packet delay on embedded systems used for networking. The single-hop one way packet delay is essential parameter when we need to process packets with strict delivery time constrains. Comparison of different approaches to single-hop one way packet delay measurements is presented in this work along with discussion about strong and weak points in specific measurement approach. The impact of different types of system load and number of CPU cores are also covered by presented results. The presented results of measurement single-hop one way packet delay in embedded Linux system show that for the specific system configuration the packet processing delay depends (in different ways) on system load and network stack load.*

Keywords

Embedded systems, packet processing delays, single-hop delay.

1. Introduction

The embedded ICT and IoT systems play more and more important role today. These systems now cover large field of applications from households to industry. As these systems became less expensive, requires less power and are equipped with more processing power it is very interesting to think about of using these devices for network packet processing even for processing time critical applications.

When processing packets on low powered embedded systems it was common that packets were preprocessed by some sort of FPGA arrays. This approach allows to process time critical network traffic in specified time constrains. With increasing processing power of modern embedded CPUs there is a question if these systems, specially the ones equipped with general purpose processor such as ARM, MIPS, x86 or other low power processors, are able to handle time critical traffic without expensive FPGA assistance. In this case not only the transmission delay is essential but even more important is delay jitter. Due to the fact that the packet processing power is very limited the single-hop one way delay measurements have its specifics on general purpose embedded systems in comparison to measurements on dedicated network platforms such as routers and switches.

We are focused on processing of packet data on embedded systems based on Linux. It is common that the processing power of these system is limited. It is quite obvious that these systems are not able to process high volume traffic but the question is where are the limits of these systems and if these systems are suitable for deployment in scenarios where on time delivery of network packets is essential.

In this article we would like to present overview of approaches and results for single-hop one way delay measurement of embedded systems such as systems based on Cavium Octeon CPU family [1]. We found Cavium hardware and software platform very interesting because it is not only a low power general purpose platform but it is also optimized for network packet processing.

2. State of the Art

The measurement of single-hop one way delay (in some literature called packet processing delay) is not a simple task especially when high precision (measurements in range of micro-seconds or even nanosecond) is needed. Single-hop one way processing delay (further we will use term single-hop delay) can be described as

a time elapsed between the moment when packet arrives to the network device t_{in} and the time when the same packet or reply to the packet is send out from the network device t_{out} Eq. (1).

$$\text{delay} = t_{out} - t_{in} \tag{1}$$

In general there are several approaches for single-hop delay measurement from some quite simple methods to sophisticated very accurate methods.

2.1. Single-Hop Measurement Approaches

1) Oscilloscope Measurement

Very precise and unique, but also very limited method of measurement of single hop delay is approach based on observation of electrical representation of single packet transferred over the wire [2] and [3]. This method when properly used is very accurate. This method is very convenient for reference measurements where it is not necessary to measure the single-hop delay under network load.

2) Packet Capture Analysis

The common methods are based on packet capture analysis. In this case it is necessary to match packets arriving into the device with packet departing the measured device. The difference between arrival and departure time stamp of packet is the processing delay of packet.

In situation where we have access to physical medium it is possible to use advanced techniques for single-hop delay measurement. It is possible to tap the probes directly to the medium connected to the measured host and obtain results from analysis of packet incoming and outgoing from interface [4].

3) OS Kernel Hooks

There is also possibility to use methods based on OS kernel hooks, which are able to trace the time the packet spends in different kernel and network driver structures [5]. This method seems to be very precise if there is a support for precise timing in OS kernel.

4) End to End Measurement

In general we can also use direct measurement based on application network response. Typical scenario is shown in Fig. 1. This approach is correct when we want to measure the whole communication chain. The possibility to use specific application protocol is an advantage. But in case we need to measure the time critical processing power of target platform the results can be misleading as we do not measure single-hop delay but we measure Round Trip Time (RTT).

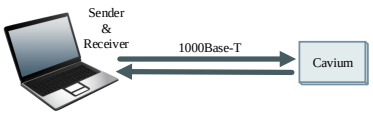

Fig. 1: End to End delay measurements - Round Trip Time.

2.2. Single-Hop Measurement Issues

The discussion over the measurement weaknesses is necessary. The presented methods from previous sections are very different and all methods have strong and weak points.

1) Time Synchronization Issue

There is a challenge of precise time synchronization on measurement host in case that the arrival packet is captured on different host than the departing packet [6] and [7]. There are several solution of time synchronization involving usage of precise time sources such as atomic clock or GPS. Another solution is based on synchronization of time among measurement hosts using computer network [8] and [9].

Also the combination of precise time source and distribution of time over the network is possible. In such case only the precise time synchronization is critical not the precise time settings itself [10], [11] and [12].

The papers [13] and [10] show the possibilities of GPS time synchronization in one way delay measurement and address the issues of this approach.It is also possible to use the GPS time source on one host and use some network synchronization method set up time on the rest of the hosts in network.

2) Packet Rate Limit Issue

The weak point of the measurement of electrical representation of single packet by oscilloscope is the fact that we can process only one packet in the window of several second (approximately 5 seconds). In this case the tested network device can not operate under real network traffic load or the network load have to be simulated through different network interface. Even in this case we do not measure only the measured device but it is necessary to consider the influence of network

path to the point where probes are attached. Another limitation of this type of measurement is the coding on the physical medium. We need coding in which we can simply distinguish when frame starts and when it ends. Because of this we are limited to 10Base-T Ethernet technology where Manchester coding scheme is used [3].

3) Packet Capture Issue

The main drawback of the method using packet capture is the fact that on the one side we can mirror traffic to be captured and analyzed with minimum delay on network switches but on the other side the final capture of packets is a real challenge on common computer. Packet capture processing is influenced by several factors such as the current load of system, the implementation of network stack, the used network interface card, the drivers used and also by the kernel settings of the system. To ensure precise time stamping of captured packets it is necessary to use special network adapters [13] and [14] that can time stamp the arriving packet otherwise this is done in network driver. Time stamping in network driver is influenced by OS setup and current system load.

4) Kernel Hooks Issues

In this case it is necessary to modify the operating system kernel which can differ on various devices. This approach has also impact on packet processing itself. The CPU power of measured host is also consumed by measurement kernel hooks. The situation is even worse, because the kernel processes are typically scheduled with higher priority than user space processes. And as we know the processing power of embedded devices is strictly limited to conform power constrains.

5) End to End Measurements Issue

The method using end to end application (real application data can be used and analyzed. Processing delay of network packets of measured system is identical with real usage of system) base measurement can provide results documenting end to end communication chain. When dealing with time critical packet processing on target system the some issues needs to be considered as in previous measurement setup and additionally the processing delay of the packet sending and receiving system should be considered.

3. Test-Bed

This paper deals with measurement of single-hop delay on Cavium OCTEON II (CN6220) platform with Linux kernel 3.10.20.

For experiments which are based on precise oscilloscope measurement on physical medium we used Tektronix DPO4032 oscilloscope. The oscilloscope maximum sampling frequency is 2.5 $GS{\cdot}s^{-1}$ which is more than sufficient for measurements on 10Base-T Ethernet. The network support in oscilloscope is an advantage for automation of data acquisition from oscilloscope.

In majority of measurements we used the Mikrotik RB2011LS-IN device for switching, bridging and port-mirroring of packets for measurement network interconnection. This device is equipped with fast switching fabric allowing to switch packets with minimum single-hop delay. The delay is mainly depended on packet size and medium speed. We can expect that single-hop delay in switching mode on 1000Base-T is below 2 µs for packet size of 192 B. The different situation is in bridged mode, where single-hop delay is significantly longer [2].

As packet generator and receiver we use our own FlowPing tool [15] and [16] capable of network stress testing based on predefined profiles. The FlowPing tool is able to stress test the network via UDP protocol in way similar to standard ICMP ping tool.

In the measurement utilizing port-mirroring we used various platforms to show the big difference in packet capture on common computer and operating systems. We involve following platform for testing: Apple MacBook Pro 13" (late 2014) with OS X 10.11 and BCM5701 over thunderbold network interface, Lenovo ThinkPad S430 with Windows 10 and RTL8168/RTL8111 family network interface, Dell Latitude E6430 with Debian 8.3 and Intel 82579LM network interface.

4. Methodology

To obtain relevant results of packet processing delays in user space we used the FlowPing tool for UDP packet processing besides of using ICMP protocol. We made this decision because we want to test the single-hop one way delay based on processing network packets in user space application. On the contrary the ICMP ECHO REPLY packets are handled in kernel space which also means that different scheduling policy can be applied on the code responsible for packet handling.

The very first measurements were targeted on system without any significant system load. Only few

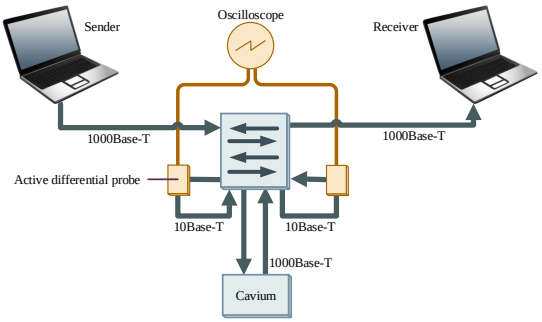

Fig. 2: Osciloscope measurement scenario - basic measurements.

packets per second were processed on target system to minimize the impact of processing of packets on system behavior. This measurements was used to set the reference value of packet processing delay on target system. We also expected that this is the lowest packet processing delay we can observe on measured system. The measurement scenario is shown in Fig. 2. In this measurement we used oscilloscope to measure processing delay of packets on target system. This approach is the most precise but have some serious limitations such as packets per second limit (we get the best results when processing 1 packet over 4–6 seconds in automatic measurement over the network connection to oscilloscope). It was also necessary to approximate the processing delay of measurement chain involving RB2011LS-IN device to create correction values for obtained results. This allow us to compute the real processing delay of target system.

Then we used several approaches to stress test target system. At first we create special tool which do nothing else than consuming CPU cycles on target system. This tool is able to operate with defined CPU load profile such as shown in Tab. 1. The described system load is fully synthetic but reflects (Profile is based on predicted behavior of expected set of services on measured system) process behavior on system with standard Linux kernel.

Tab. 1: Synthetic load generator setup.

Idle (µs)	Duty cycle (µs)	Busy-loop
77	100	False
3000	1560	False
1000	250	True
123	60	False
18000	4000	False
777	100	False
1000	1560	False
6000	250	True
1200	60	False
11000	4000	False
777	100	False
5000	1560	False
4000	250	True
1234	60	False
17000	4000	False

In this measurement we also need to force the first network controller to operate on 10Base-T. Because we need more intensive network traffic to stress test the measured system and the fact that we also need low packet rate on measured interface we decided to stress test target system with parallel network load on secondary network controller. The scenario in Fig. 3 show the measurement setup for this measurement with parallel network load.

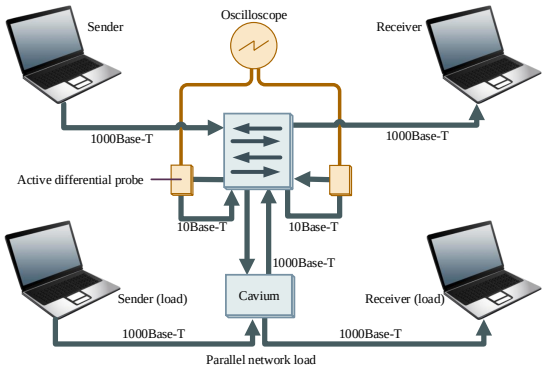

Fig. 3: Osciloscope scenario - with parallel network load.

Then we switched from measurement with oscilloscope to packet capture measurements on common computer. We used scenario shown in Fig. 4. This approach allow us to capture packets incoming on and outgoing from target system network controller. After matching corresponding packets in both directions we obtain the processing delay of packets on target system. Capture of packet via port mirroring seem to be a way to workaround the problem of the time synchronization. In this case we assume that the delay caused by packet mirroring is similar (under measurement resolution) in both TX and RX directions. That is why the real delay caused by port-mirroring is negligible. It is interesting that this method is not widely used (we did not find any mention of this method in scien-

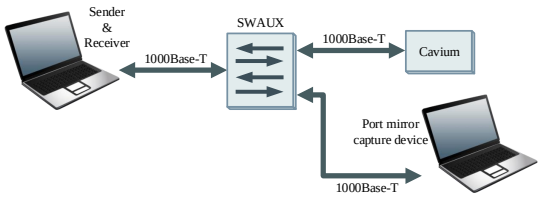

Fig. 4: Port mirroring measurements.

tific papers) in case that we have access to measured device. This method solve the issue with time synchronization of time stamps and non-standard equipment for tapping to interface is not needed. We found this method very natural and promising for evaluation of single-hop delay.

Because of the results we get from the sample measurements with port mirroring we decided to compare the results with measurements on different systems setups. For the purposes of comparison we performed this sort of test on the three common computers with different setup. More detailed description is in Subsection 5.5.

Finally we used direct method of end to end delay measurement. In this case we evaluate the direct output of FlowPing tool to measure RTT with specific traffic profile. The FlowPing tool was deployed on measured device and also on the measurement computer. Measurement scenario is shown of Fig. 1.

In the next section the most interesting results are presented. The specific size 192 B of packet used for testing purposes are not standard for benchmarking [5], but this packet size is essential in our development of new low-bandwidth protocols. And that is also why our experiments were performed with this packet size.

5. Results

5.1. Correction Values for RB2011LS-IN

As we know the bridging latency of the RB2011LS-IN device is significantly higher than in operating this device in switching mode. Because the RB2011LS-IN device is the interconnection device in majority of our measurement and especially in precise measurement with oscilloscope where this device is operating in bridge mode it is necessary to measure the bridging latency of RB2011LS-IN device to obtain correction values for further measurements. The results of measurement of correction values are summarized in Tab. 2. For the simplicity and because the Standard Deviation is not high we used only the mean value of single hop one way delay as a correction value for oscilloscope measurements Eq. (2).

$$\text{delay}_{\text{real}} = \text{delay}_{\text{measured}} - \text{correction}_{\text{mean}}. \quad (2)$$

Tab. 2: RB2011LS-IN correction values.

Mean	17.468	(µs)
StdDev	0.412	
Median	17.468	
Minimum	14.576	
Maximum	18.540	

5.2. Oscilloscope Measurements - Minimum System Load

This measurement was performed with system where only necessary processes are running. The overall system load was maintained low to allow system to process packets as fast as possible. The obtained results as presented in Fig. 5 shows that the processing delay of packet is stable with minimum jitter. We take the result of this measurements as a reference for the following measurements. We expect that the measured value is the lowest possible value of processing delay on target device.

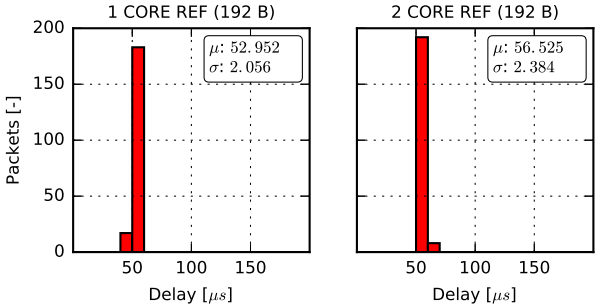

Fig. 5: Reference processing delay on load free system.

5.3. Oscilloscope Measurements - Synthetic CPU Load

This measurement demonstrates the impact of pure synthetic CPU load on packet processing. As expected the packet processing delay of packets is significantly increased in situation that CPU cores are busy. The Fig. 6 shows the processing delay of packet on system with one or two CPU cores active. The both mea-

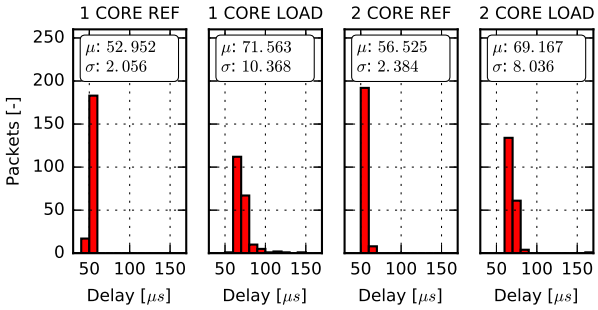

Fig. 6: Synthetic CPU load vs. number of active cores.

surements with system load show increased processing delay of packets. The results show that in case of low system load the single core system can handle packets faster but in case of high CPU load there is and advantage of using dual core system.

5.4. Oscilloscope Measurements - Parallel Network Load

Due to the fact that with increased network load the measured processing delay of packets is lower when compared to system without any load we found this measurement very interesting. As shown in Fig. 7 with increasing parallel network load on secondary network controller the processing delay of packet decreases. The main reason for this behavior is probably hidden inside OS kernel scheduling algorithm. This algorithm seem to be more effective for network stack processing with increasing number of packets processed per second.

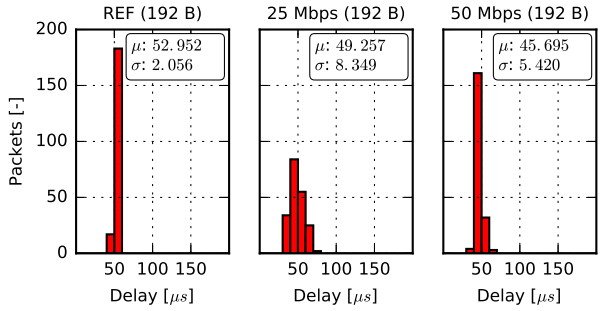

Fig. 7: Impact of processing parallel network traffic on secondary controller.

5.5. FlowPing Measurements - Packet Capture

Mirroring network traffic and capturing packets for further analysis is natural approach. But as we can see in Fig. 8 it is not an easy task to get the same results as presented in previous measurements. The reason is that if we want to capture packets on standard computer there are many obstacles. We found that even capturing traffic at rates bellow 100 Mbps on 1 Gbps network card is challenging. The presented results in Fig. 8 shows that some sort of packet buffering occur. In this case we can still get the quite precise mean value, when compared with reference measurement, of packet processing time.

But there are also several drawback. At first it is not possible to obtain the information about upper bound of processing time of packet on target device as the value we obtain is burdened by specific processing algorithm of packets in incoming queue and even worse the packet processing is also controlled by system

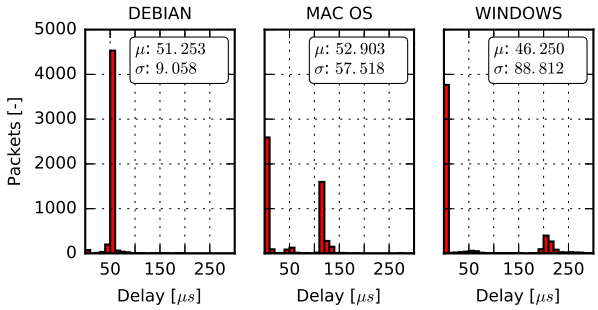

Fig. 8: Dependency of packet capture results on capturing platform setup.

scheduler. This behavior is characterized by measured delays when in some situation the measured delay is near to zero and on the other hand measured delay is several times higher that mean value. We encounter on the same problem when we want to obtain the minimum value or the jitter of processing delay.

Results show that it is necessary to equip measurement host by special packet capture network card for precise time stamping of captured packets just after their arrival on the network interface.

5.6. FlowPing Measurements - End to End

The result presented in Fig. 9 and Fig. 10 are different. In this measurement it is necessary to consider several

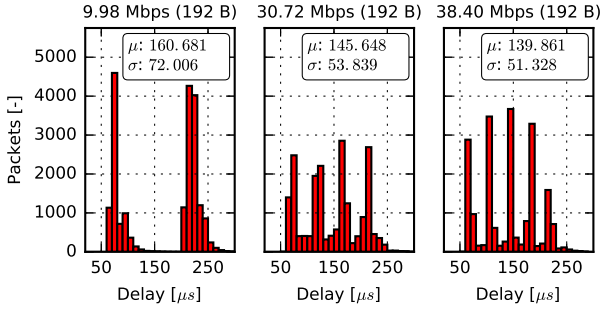

Fig. 9: Application based End to End measurements.

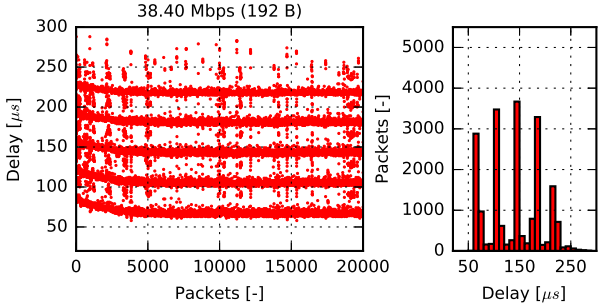

Fig. 10: Graph describing depth of queuing based on network load.

things. In this specific case we are not able to measure only the processing delay of target system but the measured processing delay is increased by the processing delay on measuring computer. Also the queuing mechanism and process scheduler have impact on measured values. The result with different amount of processed traffic shows that with increased traffic volume there is a higher probability that more packets can be present in the network queue at the same time as shown in Fig. 9. When we look at the detail of processed traffic as shown in Fig. 10 we can observe the similar behavior as observed in measurements with parallel network load. The processing delay decreases at the begging of measurement as the system is loaded by network traffic and process scheduler work more efficiently. It is necessary to mention that the duration of interval when we observer the processing delay decrease is shorter than 200 ms (aprox. 20000 packets).

6. Conclusion

Presented results demonstrates well the strong and weak points of single-hop delay measurement approaches on embedded system. We found that the measurement results from port mirroring based measurements on standard hosts are very close to measurements from method using oscilloscope when comparing the mean value of processing delay. The variance of processing packet delay differs a lot. This is caused by buffering of packets end scheduling mechanisms on measurement platforms.

We also found the some results very interesting. Especially the ones where the processing packet delay decreases with increasing amount of processed traffic. This indicates that the network stack and it scheduling is more effective under network load. On the other hand when the measured device is stressed by pure CPU load the processing delays and jitter of processing delays increases. We can assume that situation could be worse in case when the system load is not only CPU based but also the I/O system is stressed.

As we can see it is possible to obtain very accurate value of mean packet processing time. The real challenge is evaluation of the upper bound of packet processing delay when measured device is processing significant amount of network traffic. None of the presented methods and results are able to provide us the information about the real upper bound of single-hop delay under real network load. As a possible future solution we suggest to use approach with port mirroring but it is necessary to use special network controllers capable of time stamping packet just after their arrival to controller.

The measurement on application layer is valuable source of data for evaluation of application to application network latency but the impact of packet processing power, implementation of network stack and OS scheduler of the measurement host have high impact on final results. And of course in this case results does not show the single-hop delay of measured device but only the RTT value.

Acknowledgment

This work was supported by the Grant of the Technology Agency of the Czech Republic, No. TA04011571, Radio for Smart Transmission Networks, and was researched in cooperation with RACOM and Brno University of Technology. Also this work was supported by Student grant at Czech Technical University in Prague SGS16/158/OHK3/2T/13.

References

[1] OCTEON Multi-Core Processor Family. *CAVIUM* [online]. 2016. Available at: `http://www.cavium.com/OCTEON_MIPS64.html`.

[2] HEGR, T., L. BOHAC, Z. KOCUR, M. VOZNAK and P. CHLUMSKY. Methodology of the Direct Measurement of the Switching Latency. *Przeglad Elektrotechniczny*. 2013, vol. 89, no. 7, pp. 59–63. ISSN 0033-2097.

[3] HEGR, T., M. VOZNAK, M. KOZAK and L. BOHAC. Measurement of Switching Latency in High Data Rate Ethernet Networks. *ELEKTRONIKA IR ELEKTROTECHNIKA*. 2015, vol. 21, no. 3, pp. 73–78. ISSN 1392-1215. DOI: 10.5755/j01.eee.21.3.10445.

[4] SALEHIN, K. M. and R. ROJAS-CESSA. Measurement of packet processing time of an Internet host using asynchronous packet capture at the data-link layer. In: *IEEE International Conference on Communications (IEEEICC13)*. Budapest: IEEE, 2013, pp. 2550–2554. ISBN 978-1-4673-3120-3. DOI: 10.1109/ICC.2013.6654918.

[5] ANGRISANI, L., G. VENTRE, L. PELUSO and A. TEDESCO. Measurement of processing and queuing delays introduced by an open-source router in a single-hop network. *IEEE Transactions on Instrumentation and Measurement*. 2006, vol. 55, no. 4, pp. 1065–1076. ISSN 0018-9456. DOI: 10.1109/TIM.2006.876542.

[6] ALMES, G., S. KALIDINDI and M. ZEKAUSKAS. RFC 2679. *A One-way Delay Metric for IPPM.* IETF, 1999.

[7] PAXSON, V., G. ALMES, J. MAHDAVI and M. MATHIS. RFC 2330. *Framework for IP Performance Metrics.* IETF, 1998.

[8] MILLS, D., J. MARTIN, J. BURBANK and W. KASCH. RFC 5905. *Network Time Protocol Version 4: Protocol and Algorithms Specification.* IETF, 2010.

[9] SALEHIN, K. M. and R. ROJAS-CESSA. Measurement of packet processing time of an Internet host using asynchronous packet capture at the data-link layer. In: *IEEE International Conference on Communications (ICC).* Budapest: IEEE, 2013, pp. 2550–2554. ISBN 978-1-4673-3122-7. DOI: 10.1109/ICC.2013.6654918.

[10] Precise Measurement of One-Way Delay and Analysis of Synchronization Issues. *CiteSEER* [online]. 2002. Available at: `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.14.3218&rep=rep1&type=pdf`.

[11] PRAVDA, M., J. VODRAZKA and P. LAFATA. Simulations and Measurements of Packet Network Synchronization by Precision Time Protocol. In: *35th International Conference Telecommunications and Signal Processing.* Prague: IEEE, 2012, pp. 116–120. ISBN 978-1-4673-1117-5. DOI: 10.1109/TSP.2012.6256264.

[12] LOESER, J. and H. HEARTIG. Low-latency hard real-time communication over switched Ethernet. In: *16th Euromicro Conference on Real-Time Systems 2004 (ECRTS 2004).* Catania: IEEE, 2004, pp. 13–22. ISBN 0-7695-2176-2. DOI: 10.1109/EMRTS.2004.1310992.

[13] PAPAGIANNAKI, K., S. MOON, C. FRALEIGH, P. THIRAN and C. DIOT. Measurement and analysis of single-hop delay on an IP backbone network. *IEEE Journal on Selected Areas in Communications.* 2003, vol. 21, no. 6, pp. 908–921. ISSN 0733-8716. DOI: 10.1109/JSAC.2003.814410.

[14] (DAG) Data Capture Cards. *Endance* [online]. 2016. Available at: `https://www.endace.com/endace-dag-high-speed-packet\-capture-cards.html`.

[15] VONDROUS, O., P. MACEJKO and Z. KOCUR. FlowPing - The New Tool for Throughput and Stress Testing. *Advances in Electrical and Electronic Engineering.* 2015, vol. 13, no. 5, pp. 516–521. ISSN 1804-3119. DOI: 10.15598/aeee.v13i5.1497.

[16] VONDROUS, O., P. MACEJKO and Z. KOCUR. FlowPing Application. In: *Czech Technical University in Prague, Faculty of Electrical Engineering.* [online]. 2013. Available at: `http://flowping.fel.cvut.cz`.

About Authors

Ondrej VONDROUS was born in Czech Republic in 1981. He received his M.Sc. degree in electrical engineering from the Czech Technical University in Prague in 2011. Since 2011 he has been studying Ph.D. degree in telecommunication engineering. His research interests include network transmission control, data flow analysis and data flow optimization.

Peter MACEJKO was born in Czech Republic in 1980. He received his M.Sc. degree in electrical engineering from the Czech Technical University in Prague in 2006. He is teaching networking technologies and distributed systems. His research is focused on scheduling in distributed systems and data flow and protocol analysis. He is currently actively involved in projects focused on high speed data transmission from fast moving objects.

Zbynek KOCUR was born in 1982. He received his M.Sc. degree in electrical engineering from the Czech Technical University in Prague in 2008 and Ph.D. degree in electrical engineering in 2014. He is teaching communication in data networks and networking technologies. His research is focused on wireless transmission and data flow analysis, simulation and optimization.