Performance Evaluation of the Nanosecond Resolution Timestamping Feature of the Enhanced Libpcap

Peter Orosz, Tamas Skopko, Jozsef Imrek
Faculty of Informatics
University of Debrecen
Debrecen, Hungary
e-mail: oroszp@unideb.hu, skopkot@unideb.hu, mazsi@unideb.hu

***Abstract* — In a previous work we modified the libpcap library in order to feature nanosecond resolution timestamping. However the precision and the accuracy of this high resolution software based solution has not been investigated so far. Since very short inter-arrival times are present on Gigabit Ethernet links, the precision of the software based timestamps generated for each incoming packet should be analyzed and validated. In this paper, the performance metrics of the nanosecond resolution software timestamping is compared to a hardware-based solution (Endace DAG3.7GP measurement card) and to a reference source. The factors that determine and limit the precision and accuracy of the nanosecond resolution software timestamping will be investigated. We present how the operation mode of the network device driver affects the timestamping behavior.**

Keywords-libpcap; timestamp precision; inter-arrival time; Linux kernel; high speed network; hardware timestamping.

I. Introduction

Measurement of high performance networks requires high performance timestamping [1][2]. Using libpcap based capturing limits both the resolution and the precision of the generated timestamps [3]. Although in a recent work we enhanced the timestamping resolution of the original libpcap library [4], it is hard to improve the precision of the nanosecond resolution software based timestamping to meet the needs of the Gigabit Ethernet and faster networks. Even with the most sophisticated optimization, the precision of the software timestamp could not compete with the hardware based one [5]. In this paper, we will investigate the precision of the high resolution timestamping feature of the enhanced libpcap library. A widely used high performance Gigabit Ethernet NIC (Network Interface Card) and driver combination was selected for our measurements. According to the result of the investigation, the question arises with good reason: Is there any measurement type where the application of nanosecond resolution software timestamping is reasonable? To answer this question, we performed comparative measurements: one test setup was assembled for fixed packet size generated at high, fixed rate and the other one for replayed VoIP traffic including variable packet size. Measurement with fixed packet size and inter-arrival time is suited to analyze the deviation of inter-arrival times presented by the generated timestamps. Whereas replayed traffic consisting of packets with variable sizes is adapted to show how precisely the high resolution software timestamping could represent traffic characteristics.

II. Measurement Setup

In both measurements two independent hosts were used: one for software timestamping and the other one for reference hardware timestamping, with both systems simultaneously capturing the same traffic. In every measurement we made sure that no packet loss occurred.

A. *Line-rate traffic generator*

In our first measurement setup a NetFPGA-1G card [6] was used for mirroring the ingress traffic on its PORT A to two of its Gigabit Ethernet ports (PORT B and PORT C). An FPGA (Field Programmable Gate Array: Xilinx Virtex-4 FX12) based configurable line-rate packet generator device was connected to the PORT A of the NetFPGA-1G board. PORT B was connected to an Endace DAG 3.7GP monitoring card (installed in a dedicated host) [7], and PORT C was connected to an Intel PRO/1000 PT Gigabit Ethernet network interface card (Fig. 1) [8].

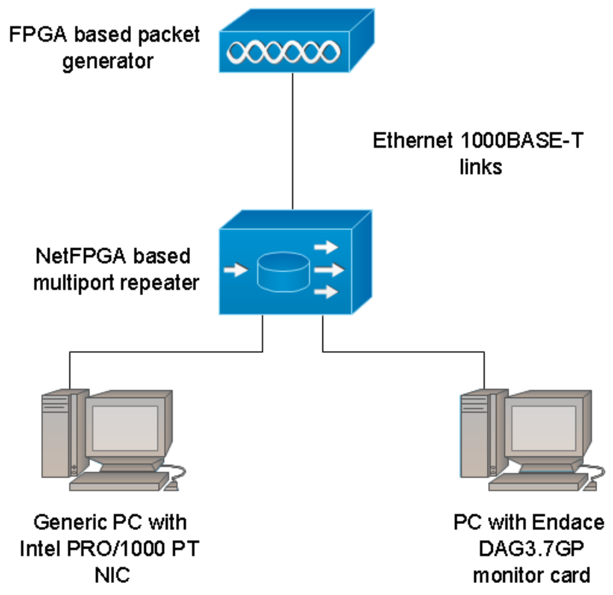
FPGA based packet generator
Ethernet 1000BASE-T links
NetFPGA based multiport repeater
Generic PC with Intel PRO/1000 PT NIC
PC with Endace DAG3.7GP monitor card

Figure 1. Layout of the first measurement setup

The packet generator was controlled by UDP packets using a proprietary protocol sent from the NetFPGA-1G host. Our NetFPGA-1G loopback module makes it possible to directly loop back certain ports of the board to other ones with a minimal, fixed amount of latency, using no store-and-forward mechanisms.

In this configuration, any traffic received by PORT A is directly forwarded to PORT B and C but not the traffic transmitted on PORT A (which was used to send control packets to the generator). Software timestamps were created based on a TSC clock source by capturing on the Intel PRO/1000 NIC, and Endace's hardware timestamping feature was used to create reference timestamps on the DAG board [9].

B. *NetFPGA-1G multiport packet generator*

In the second setup, we utilized an existing NetFPGA.org project ("Packet generator" [10]) as to perform another measurement series using variable packet size. This generator can replay any previously recorded stream of packets stored in PCAP format. In this setup, the NetFPGA's PORT B was connected to the Endace DAG3.7GP and its PORT C was connected to the Intel PRO/1000 NIC (Fig. 2).

For these measurements the Packet Generator code was loaded into the NetFPGA-1G board, and it was configured to send the same traffic stream on both PORT B and PORT C. Similarly to the previous configuration, the host with the Intel PRO/1000 NIC captured packets with software timestamps based on the system's TSC clock source and the other host with the DAG3.7GP board provided the reference hardware timestamps.

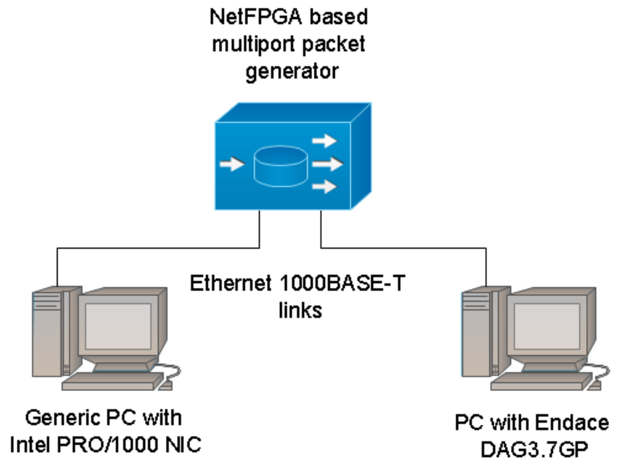
NetFPGA based multiport packet generator
Ethernet 1000BASE-T links
Generic PC with Intel PRO/1000 NIC
PC with Endace DAG3.7GP

Figure 2. Layout of the second measurement setup

All of the measurements were performed on a non-preemptive 2.6.35.7 Linux kernel.

III. MEASUREMENT RESULTS

It is important to note that hardware timestamps show inter-arrival times of the packets as seen in the MAC layer. In contrast, the software based timestamps represent the time of the enqueuing of the received packets into the Linux kernel's input packet queue [11].

Nevertheless, there is an obvious difference between the hardware and software timestamps in absolute time since their generation occurs at different points of the data path.

Parameter settings of the Intel e1000e Linux driver for the Intel PRO/1000 NIC family affect the kernel-level processing of Ethernet frames, and therefore the generation of software timestamps. The interrupt-handling parameters of the driver can be tuned at module loading. The increasing rate of the received packets increases the rate of the interrupts as well, and this increased number of interrupts can be served by the processor only up to a certain threshold value. Above this threshold more frames should be transmitted to the kernel within one interrupt to maintain the lossless packet reception.

The *InterruptThrottleRate* parameter of the driver and the incoming packet rate together defines the timing of frame processing. When the value of *InterruptThrottleRate* is 0, there is no critical value for the interrupt rate above, which the driver would explicitly decrease their number by transmitting more frames from the card to the kernel during an interrupt (i.e., no coalescing occurs). This is the classic one frame per interrupt mode, which works well with low packet rate and provides low latency, but leads to the exhaustion of hardware resources when traffic load is heavy. To avoid this exhaustion dynamic modes can be used: *InterruptThrottleRate=1* and the conservative *InterruptThrottleRate=3* modes. For more details on the interrupt throttling modes of the Intel E1000E Linux driver, see [12].

The measurements were performed in all three modes of the Intel device driver. The advantage of the nanosecond resolution timestamping recurs significantly with low packet inter-arrivals (i.e., high packet rate and small packet size). However, the nanosecond resolution does not guarantee precision with a similar magnitude at all. Besides that the generation cost of software timestamps in CPU time is significantly higher than the hardware timestamp's production, the length of generation time can display notable fluctuations. The main reason for this can be traced back to shared hardware resources. The timestamp is created in the kernel context, the execution of the producing function call series are being scheduled similarly to every other kernel process. Furthermore, the process of the timestamp-creation can be suspended, e.g., by a hardware interrupt. These factors all contribute to the apparent high fluctuations of inter-arrivals represented by the timestamps. In order to avoid preemption of the execution of the timestamping kernel code, all of our measurements were performed using a non-preemptive Linux kernel.

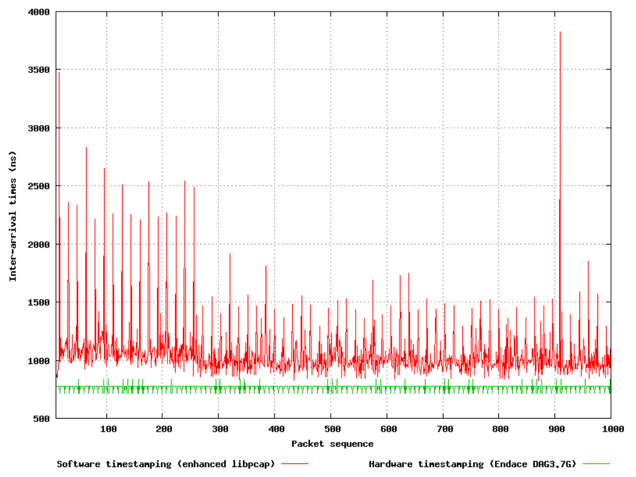

Figure 3. Packet inter-arrival times for IFG=12 bytes and InterruptThrottleRate=0. Green line represents the inter-arrival times measured by the Endace DAG board, while the red line shows the inter-arrival times measured by the enhanced libpcap.

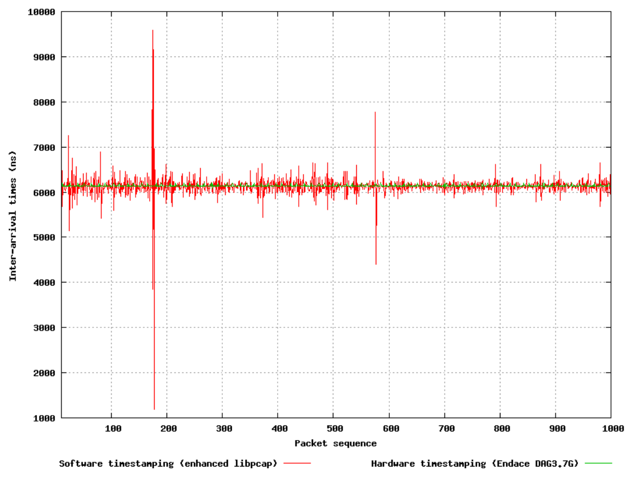

Figure 4. Packet inter-arrival times for IFG=684 bytes and *InterruptThrottleRate=0*

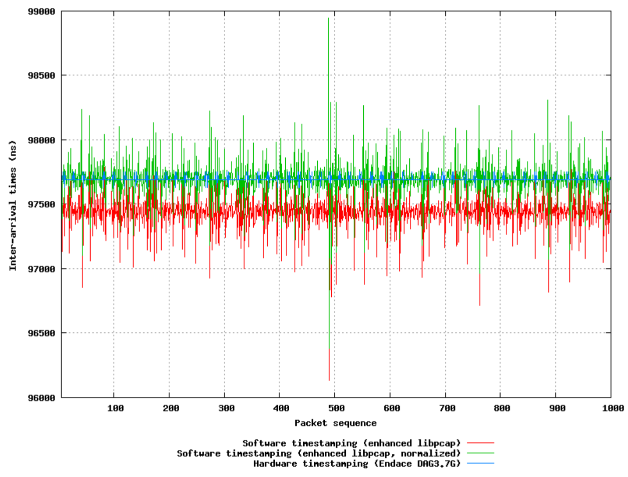

Figure 5. Packet inter-arrival times for IFG=12188 bytes and *InterruptThrottleRate=0,* normalized software timestamps

The traffic generator used in the first measurement environment generates line-rate packet sequences with a fixed packet size and IFG (inter-frame gap) value. Due to the traffic being generated in hardware, the inter-arrival times of the packets are constant. The FPGA traffic generator's clock signal is 125 MHz (nominal), resulting in a 8 ns clock signal period. In case of Gigabit Ethernet network the minimum of packet inter-arrival times is 672 ns.

During the measurements a fixed packet size (72 bytes, excluding the 8-byte preamble and 4-byte CRC fields) was used together with the following IFG values: 12, 684 and 12128 bytes. In fact, the packet size and the IFG value together define the arrival rate, which, according to our presumptions directly affects the Intel NIC driver's operation.

Every measurement was carried out using traffic data consisting of 1000 packets with the parameters described above. During the measurements the focus was on the arrival intervals of the packets, thus relative timestamps were used.

The resolution of the timestamps generated by the Endace DAG3.7GP monitor card is 60 ns. The inter-arrival times indicated with green on the figures were calculated on the bases of hardware timestamps.

Figure 2, 3, and 4 show a ±60 nanosecond (±1 clock) deviation of the packet arrival-intervals, which is caused by the fact that the traffic generator and the Endance measurement card are running on asynchronous (unrelated) clocks.

A. *Packet reception with no interrupt throttling*

When the Intel e1000e driver runs with *InterruptThrottleRate=0* parameter, it does not apply interrupt moderation. The arrival intervals calculated from the software timestamps showed large deviation regardless of the measurements, they could not produce the estimated 672 ns Δt values in case of 12-byte IFGs as expected (Fig. 3) [13][14]. The reason for this is that it takes more time to generate the software timestamps than the inter-frame arrival times. The short burst of the 1000 packets used for the measurement was stored in the device driver's ingress queue (in the DMA buffer area allocated by the driver), and the packets were moved into the input packet queue of the Linux kernel's network stack in a pace determined by the execution time of the timestamp generation and other additional housekeeping duties associated with packet reception.

As it clearly shows on Figure 3, the Intel e1000e driver has an adjustment period (the first 260 packets, approx.) according to which it configures the interrupt parameters. Thereafter the deviation of the timestamps is reduced significantly; however, it is not at all free of unexpected peaks (around the 900th packet). Each of the inter-arrival times derived from the software timestamps represents higher values compared to the hardware based arrival times, which could lead to packet loss in a long term measurement.

This phenomenon is almost impossible to eliminate in case of software timestamping due to the kernel pacing and the interruptions during the course of operation. The frequency of these kinds of peaks increases during longer measurement periods. In the second measurement (IFG=684 bytes) the mean value of the inter-arrival times based on software timestamps aligns with the values calculated on the basis of hardware timestamps, but their deviation magnitude lies within the microsecond-range (Fig. 4). The third measurement shows similar results to the second one in terms of timestamp deviation, however the mean values has an obvious difference (Fig. 5). The reason for this can be traced back to the fact that the host with software timestamping and the Endace measurement card are running on asynchronous (unrelated) clocks.

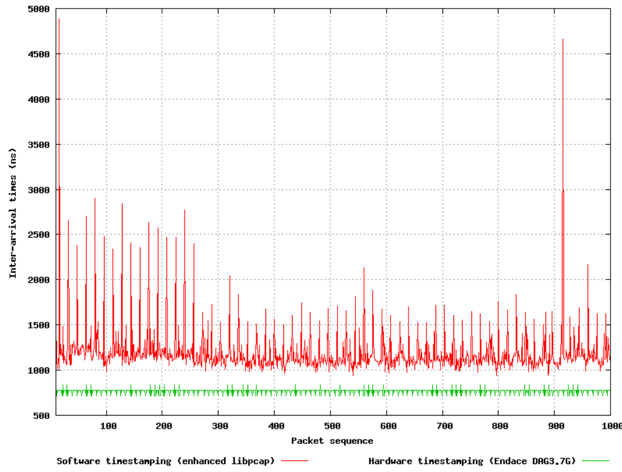

Figure 6. Packet inter-arrival times for IFG=12 bytes and *InterruptThrottleRate=1*

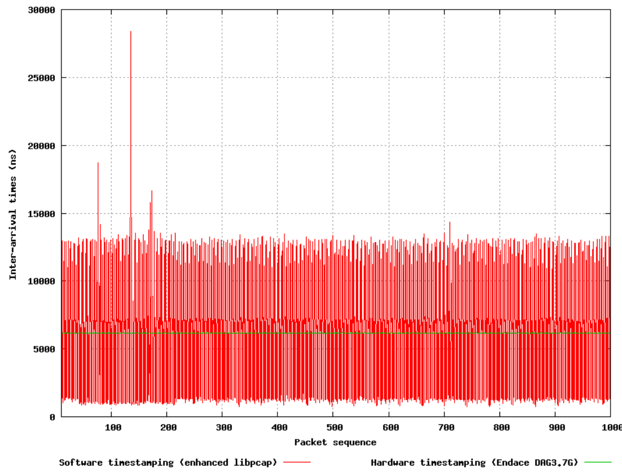

Figure 7. Packet inter-arrival times for IFG=684 bytes and *InterruptThrottleRate=1*

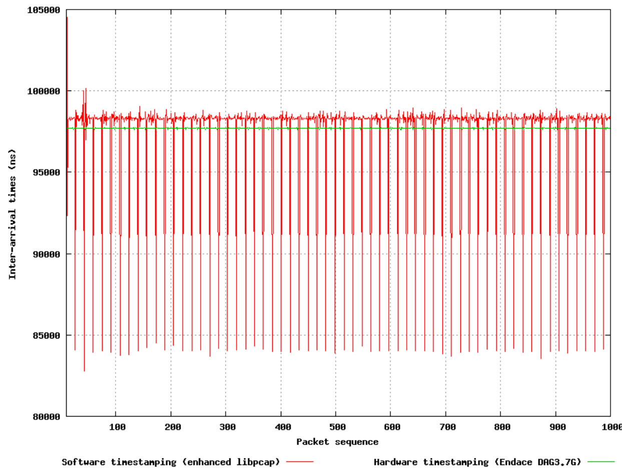

Figure 8. Packet inter-arrival times for IFG=12188 bytes and *InterruptThrottleRate=0*

Accordingly, the Endace DAG hardware clock was designated for reference clock that the software timestamp sequences have to be normalized to. We calculated the average of the Δti (time difference between the arrival of the ith and 1000 – 50 + ith, where i=0…49) for both the software and hardware packet sequences. The quotient of the averages resulted in a normalizing constant, which was used to correct the software based timestamps (Fig. 5).

B. *Packet repection with dynamic interrupt throttling*

When the *InterruptThrottleRate* parameter of the e1000e driver is 1, the driver adjusts to the incoming traffic rate by dynamically tuning the value of the interrupt moderation (polling mode).

In case of high arrival rate (IFG=12 bytes) the precision of the software timestamps deteriorates compared to the zero-throttle mode used in the previous measurement series (Fig. 6).

The advantage of the polling mode in case of heavy traffic is the lower hardware resource requirement. In case of polling mode the inter-arrival times defined by software timestamps reflect the polling mechanism's frame enqueuing rate and timing rather than the temporal relation experienced on the MAC level.

At lower arrival rate the results gathered from the hardware and software timestamping are significantly different. With higher IFG it is more conspicuous that due to the polling-based processing the timestamps reflect the enqueueing time intervals of the incoming frames rather than the MAC-level time intervals (Figs. 7, 8). The question emerges whether the high resolution (1 ns) context-dependent timestamping mechanism combined with a polling-based NIC driver can be used for high speed traffic analysis.

For measuring the time relation of the frames as seen in the MAC layer the right solution is clearly the high resolution and precision hardware timestamping. However, if we are interested in the frame arrival interval to the kernel, the relevant information can be gathered from the software timestamps produced by the kernel.

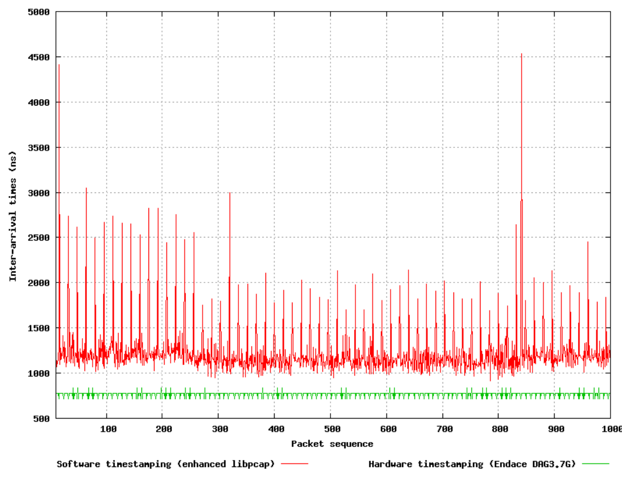

Figure 9. Packet inter-arrival times for IFG=12 bytes and *InterruptThrottleRate=3*

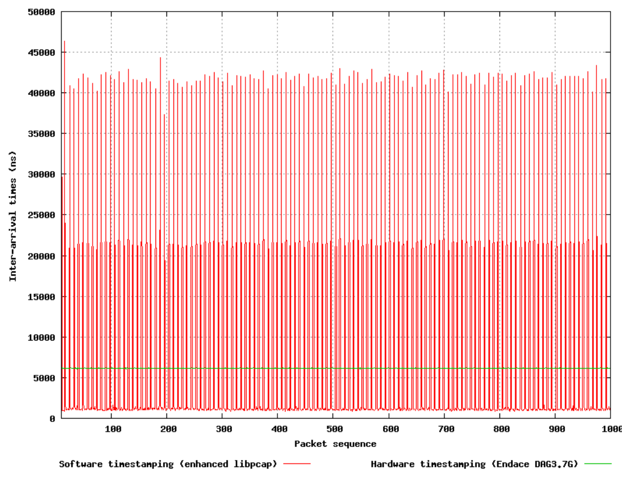

Figure 10. Packet inter-arrival times for IFG=684 bytes and *InterruptThrottleRate=3*

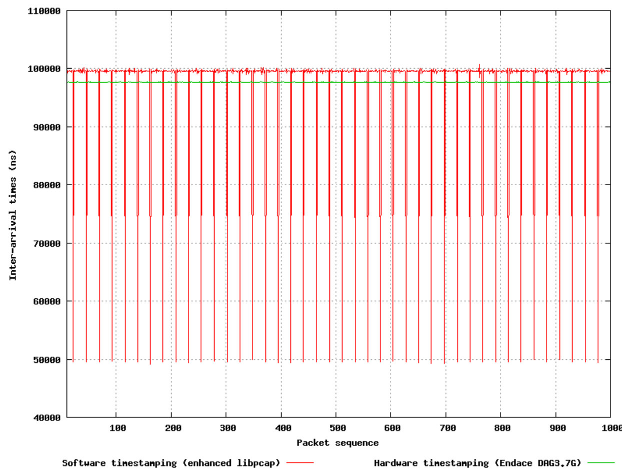

Figure 11. Packet inter-arrival times for IFG=12188 bytes and *InterruptThrottleRate=3*

C. *Packet reception with conservative dynamic interrupt throttling*

The results of the measurements with *InterruptThrottleRate=3* settings show that the lack of low latency processing significantly increases the deviation of the measured arrival intervals (Fig. 9).

The resulted maximum of arrival intervals develops between those consecutive frames, which were processed in two time adjacent interruption contexts. Meanwhile, due to the polling mechanism the arrival interval of the frames processed consecutively during one interrupt are much closer to each other. This low value is derived solely from the host's processing latency: enqueueing and timestamping (Figs. 10, 11). Nevertheless, contrary to the conventional microsecond resolution the high resolution software timestamping of the packets can represent the arrival moment of the packets to the operation system with higher accuracy.

D. *Replayed VoIP traffic*

In the second round of the investigation a measurement environment was created where arbitrary, previously recorded traffic samples could be replayed, while preserving the original packet inter-arrival times.

Variable packet size was the primary aspect in choosing the PCAP traffic sample. The aim of this measurement series was to examine the accuracy of the high resolution software timestamps generated by replaying and capturing a real VoIP traffic sample (RTP transmission with G.711 audio data).

As it is apparent from the results of measurements performed with *InterruptThrottleRate=0* settings, even though the arrival intervals defined by software timestamps have larger deviation, in this case they can reflect the time relation denoted by the hardware timestamp sequence (Fig. 12).

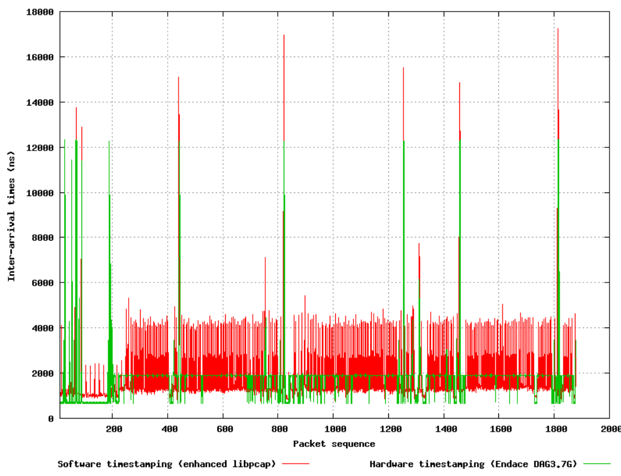

Figure 12. Packet inter-arrival times for *InterruptThrottleRate=0* (variable packet size)

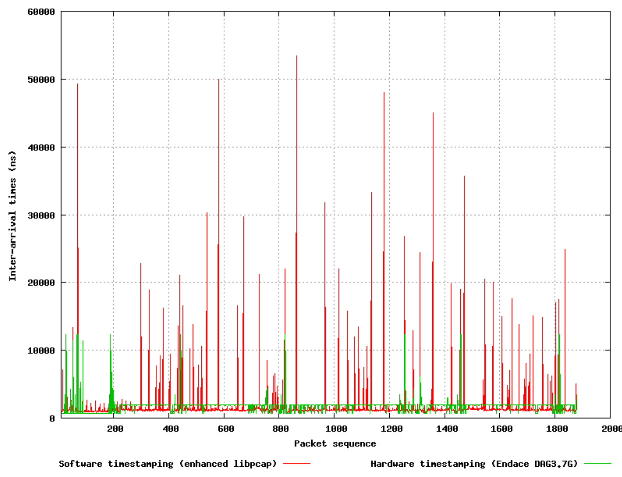

Figure 13. Packet inter-arrival times for *InterruptThrottleRate=1* (variable packet size)

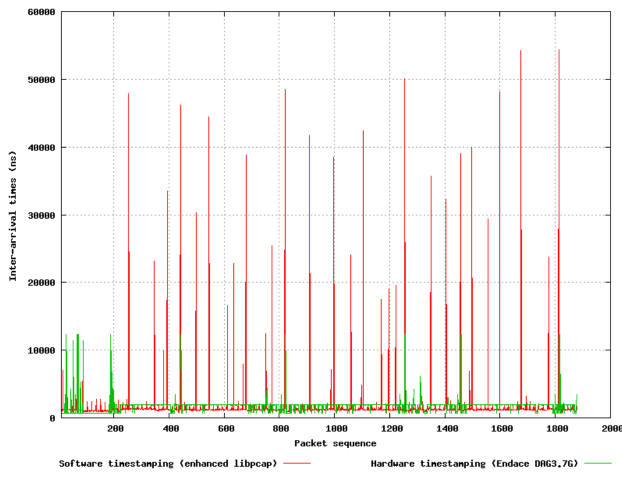

Figure 14. Packet inter-arrival times for *InterruptThrottleRate=3* (variable packet size)

Due to the effect of interrupt moderation and polling applied in dynamic (*InterruptThrottleRate=1)* and conservative (*InterruptThrottleRate*=3) modes, the packets are added to the kernel's input packet queue according to the process pacing, thus their MAC-level temporal relation is lost (Figs. 13, 14).

IV. Conclusion

At high arrival rate, to measure the time relation of the incoming frames as seen in the MAC layer the right solution is the high resolution and precision hardware timestamping. However, if we are interested in the arrival interval of the frames to the kernel network stack, then the relevant information can be obtained from the high resolution software timestamps produced by the kernel. At low arrival rate the results derived from software timestamps could come close to the ones represented by hardware timestamps. Besides that the generation cost of software timestamps in CPU time is significantly higher than the hardware timestamp's production, the length of generation time can display notable deviations. A possible solution to mitigate this problem is to store the raw value of the time reference as a timestamp (i.e., the value of the TSC counter), and convert it into time of the day (i.e., nanoseconds) format offline.

Nevertheless, contrary to the conventional microsecond resolution, the high resolution software timestamping of the packets can represent the arrival moment of the packets to the operation system with higher accuracy.

Acknowledgment

The work was supported by the TÁMOP 4.2.1./B-09/1/KONV-2010-0007 project. The project was implemented through the New Hungary Development Plan, co-financed by the European Social Fund and the European Regional Development Fund.

References

[1] Jörg Micheel, Stephen Donnelly, and Ian Graham, "Precision timestamping of network packets," Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement, November 1-2, 2001, San Francisco, California, USA

[2] Gianluca Iannaccone, Christophe Diot, Ian Graham, and Nick McKeown, "Monitoring very high speed links," Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement, November 1-2, 2001, San Francisco, California, USA

[3] Attila Pasztor and Darryl Veitch, "PC based precision timing without GPS," Proceedings of the 2002 ACM SIGMETRICS international conference on Measurement and modeling of computer systems, June 15-19, 2002, Marina Del Rey, California, USA

[4] Peter Orosz and Tamas Skopko, "Performance evaluation of a high precision software-based timestamping solution for network monitoring," the International Journal on Advances in Software, ISSN 1942-2628, in press.

[5] Peter Orosz and Tamas Skopko, "Timestamp-resolution problem of traffic capturing on high speed networks," January 28-30, 2010, ICAI international conference, Eger, Hungary

[6] NetFPGA, http://www.netfpga.org/, 23/07/2011

[7] The DAG project, http://dag.cs.waikato.ac.nz, http://www.endace.com, 23/07/2011

[8] Intel PRO/1000 PT NIC, http://www.intel.com/products/server/adapters/pro1000pt-dualport/pro1000pt-dualport-overview.htm, 23/07/2011

[9] TSC, Intel 64 and IA-32 Architectures Software Developer's Manual, http://developer.intel.com/Assets/PDF/manual/253667.pdf, 23/07/2011

[10] Packet generator, NetFPGA.org, http://netfpga.org/foswiki/bin/view/NetFPGA/OneGig/PacketGenerator, 23/07/2011

[11] Christian Benvenuti, Understanding Linux Network Internals, O'Reilly, 2006

[12] Intel E1000E driver documentation.

[13] IETF RFC2679, A one-way delay metric for IPPM, http://www.ietf.org/rfc/rfc2679.txt, 23/07/2011

[14] IETF 3393, IP Packet Delay Variation Metric for IPPM, http://www.ietf.org/rfc/rfc3393.txt, 23/07/2011