Thermal Effects on Timestamping in 100G Networking Devices

Stefan Lachnit, Sebastian Gallenmüller, and Georg Carle
Chair of Network Architectures and Services
School of Computation, Information and Technology, Technical University of Munich, Germany
Email: {lachnit|gallenmu|carle}@net.in.tum.de

***Abstract*—Latency and jitter are key properties of computer networks. High-speed network connections transmit packets in the order of nanoseconds, which makes latency measurements a challenging task. In this paper, we evaluate the accuracy of hardware timestamping results for three 100G networking devices: A programmable P4 switch, a commodity NIC, and an FPGA-based capture card. Our measurements focus on the effects of temperature on the measurement accuracy. Therefore, we created a setup that distributes the same optical signal to all devices at the same time, allowing a direct comparison of the observed timestamps between the three timestamping devices. The evaluation demonstrates that 100G hardware is sensitive enough to measure effects on a nanosecond timescale caused by temperature fluctuations of 1° K.**

***Index Terms*—timestamping, measurement, high-speed networks**

I. Introduction

Precise information on packet timing and, specifically, latency is essential in understanding the behavior and performance of network devices and packet processing systems. This is especially important at high data rates, e.g., for 100G Ethernet, where the serialization time for a minimum-sized packet (64 B) is only 5.12 ns. To gain insights into the behavior of 100G systems and analyze effects like packet bursts, timestamps with high precision and accuracy are required. However, measuring at nanosecond timescale reliably and consistently is challenging, especially for software-based systems. To achieve this goal, we need hardware-supported timestamping. In this paper, we investigate three different devices that claim to support 100G timestamping, a commodity Intel NIC, a specialized packet capture card based on an FPGA, and a switch based on an Intel Tofino ASIC.

To perform measurements using one of these devices requires a profound knowledge about the precision of the measurement equipment. We want to be able to precisely measure the latency of high-speed networking devices on commodity hardware without impacting the measurement results. To interpret the measured results, we need to estimate the inaccuracies introduced by the measurement instruments themselves. One known factor that influences the speed of digital clocks is temperature. Therefore, we briefly introduce hardware timestamping devices, explain our measurement methodology, and finally investigate the effect of temperature changes on the timestamping precision of the tested devices.

Most existing research focuses on the accuracy and precision of timestamping for networks up to 10G and comparisons to software-based timestamping [1], also including active Ethernet taps [2] [3]. For high-speed networks, related work mainly assesses timestamping functionality for a specific use case (e.g., time synchronization [4], or latency measurement [5] [6]) and evaluates general timestamping accuracy without considering specific effects such as temperature changes.

II. Background

Many modern commodity NICs and networking appliances support timestamping of received and transmitted packets in hardware. To minimize the deviation between the measured and the actual time of transmission, timestamping happens as close to the physical transmission as possible. This is often implemented to support precise time synchronization using the Precision Time Protocol (PTP). Emmerich et al. [1] demonstrated that this process can be used to timestamp specific packets in network flows.

These timestamps are typically sampled from an independent timer running on the networking devices. This timestamping mechanism consists of two registers: A register holding the current time and a second register storing an increment value. For every clock cycle of an oscillator on the device, the time register is incremented by the increment value. The increment value is selected based on the speed of the oscillator, so the least significant bits of the timer and increment value can be interpreted as a subnanosecond part, while the other bits represent a nanosecond part. When a packet is received, a timestamp is determined by sampling the current value of the timer register.

This functionality can also be used for precisely measuring the latency of a tested networking device. There are two main ways this can be achieved:

A. TX and RX timestamps

In this setup, depicted in Figure 1, the first packet timestamp (TS_{tx}) is captured when the packet is transmitted by the packet generator. The packet is then sent to the tested device and returned to the generator, where a second timestamp (TS_{rx}) is captured as soon as the packet is received. Based on the two timestamps, the latency of that packet caused by the Device under Test (DuT) can be calculated. To measure latency with

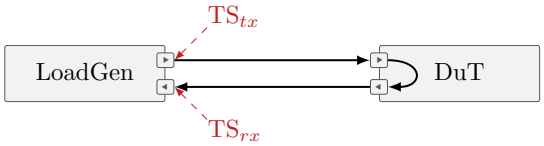

Fig. 1: Latency measurements using RX and TX timestamps

high precision, it is important to capture the timestamps late in the transmission and early in the reception process.

Such a measurement setup requires a combined traffic source and sink, which lowers the complexity of the setup and the effort to synchronize timestamps between the source and sink. However, this measurement setup has limitations. The time of transmission is unknown when a packet is handed over from the packet generator to the network stack. The timestamping hardware stores timestamps of selected transmitted packets in device registers. These registers need to be read by the packet generator, before the next packet can be timestamped, to prevent the previous timestamp from being overwritten. Consequently, this approach limits the rate of packets for which latency can be measured. Measurements show that approx. 1000 packets per second (pps) can be timestamped using the PTP registers of an Intel X520 NIC [1].

B. RX Timestamps Only

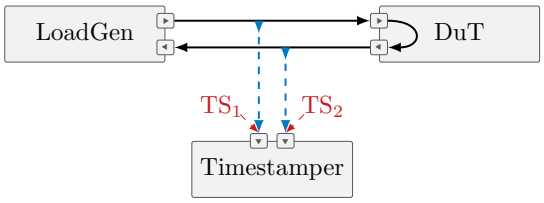

Fig. 2: Latency measurements using RX timestamps and a passive optical splitter

In addition to sampling timestamps for selected packets, some network devices also support timestamping of all received packets. The advantage of this approach—a packet generator does not need to query the timestamps on the RX path separately; the timestamps are typically accessible in software as a part of the regular packet metadata. This way of timestamping can be used to measure the latency of all packets on a link without influencing the communicating devices and limiting the timestamping process as described in Subsection II-A.

To determine the transmission timestamps, the measurement setup needs to be modified. This modified hardware setup is depicted in Figure 2. A load generator sends traffic to a DuT that returns it back to the load generator. All links are using fiber optic transceivers. Using a passive optical splitter, traffic from a single source can be sent to two devices simultaneously. This is used to additionally direct all packets sent from the load generator to the DuT on the first link and all packets returned from the DuT on the second link to a capturing host. The first link generates the TX timestamps, the second link the RX timestamps. The two capturing links are connected to a network device with two interface ports and a shared timer or with synchronized timers between its ports. Because the splitter works passively and does not influence the timing of the optical signal, measurements using this approach do not influence the tested devices. Distributing the optical signal to several ports will decrease its signal strength. However, the used long-distance transceivers have enough reserves to compensate for this loss. Using this approach, it is possible to capture timestamps for every packet at a rate of 100G using commodity hardware, because only receive timestamps are used, which can be determined for every received packet. A similar setup for 10G networks was also used in [7].

III. Evaluation

To quantify the effect of temperature on the timestamping of the tested devices, we performed an experiment on the hardware setup depicted in Figure 3. The experiment is executed in our testbed, managed by pos [8].

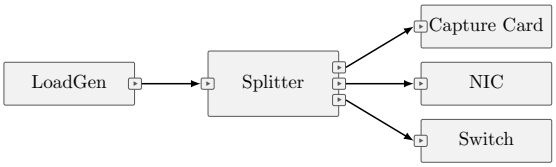

Fig. 3: Hardware setup

The load generator runs on a host with an AMD EPYC 7542 with 512 GB of RAM and an Intel E810-CQDA2 NIC.

We compare three devices of different classes: A P4 programmable switch (Intel Tofino 1), a commodity 100G NIC (Intel E810-CQDA2), and an FPGA-based packet capture card (Silicom FB2CGG3). All three devices support 100G Ethernet and offer hardware timestamping. The NIC and capture card are connected to a single capturing host, while the P4 switch adds the captured timestamp to the packet payload and forwards it to a 100G NIC on the capturing host that extracts the timestamp information. The capturing host was running on a dual-socket AMD EPYC 7542 system with 1024 GB of RAM using Intel E810-CQDA2 NICs.

We generated line rate traffic at 100 Gbit/s with a packet size of 9000 B using the MoonGen traffic generator [1]. Because frames are sent back-to-back, the expected spacing between subsequent packets is constant; its calculated length is 721.6 ns.

The generated traffic is sent through a passive optical splitter to all three tested devices, each using 1.5-m cables. This ensures that the optical signal arrives at the three devices simultaneously, allowing for direct comparisons. Timestamps are captured on all three devices for fixed bursts of received packets, selected by hardware filters, and stored in combination with packet IDs added by the traffic generator. Timestamps are stored as integers, representing the time in nanoseconds since the start of the experiment. Using the measured duration between consecutive packets (determined

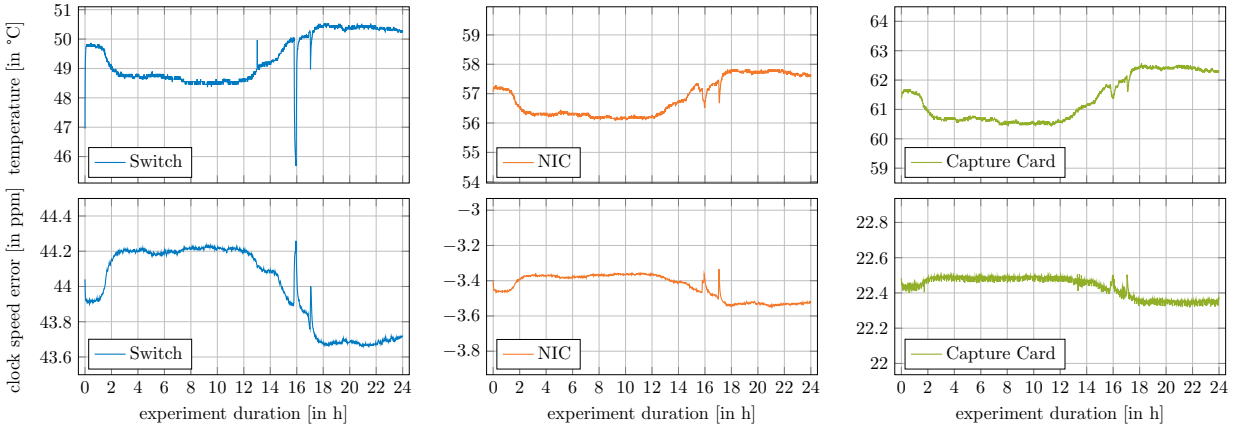

Fig. 4: Clockspeed deviation using 9000 B Packets at a line rate of 100 Gbit/s for varying QSFP28 temperatures

by the packet IDs) and comparing it with the expected spacing, the deviation of the timestamping clock from the expected clockspeed is calculated.

The experiment was executed for a timespan of 24 h while recording the temperature reported by the QSFP28 modules of the capturing devices. In total, 467 697 664 timestamps were captured for each device.

The three graphs at the top of Figure 4 show the temperature measured by the QSFP28 transceivers during the experiment, with the same vertical range. The time since the start of the experiment is shown on the x-axis. This temperature is affected twofold: After starting the experiment, the temperature increases for all three devices. This is caused by the additional load and, consequently, heat created when running the experiment. The increase in temperature is largest with the Tofino switch because our measurement scripts deactivate the transceivers for this device when not in use, while the transceivers remain active for the other two devices. After the initial increase in temperature, the changes are mostly caused by variations in the ambient temperature of the room. The identical change in ambient temperature affects devices differently because of differences in the cooling of the respective devices.

For all three devices, the temperature shows two anomalies with a sharp decrease in temperature after 16 h and 17 h, respectively. These anomalies were caused by opening the door of the server room. The sharp increase in temperature visible only for the switch at 12.5 h was caused by a measurement error when reading the temperature from the QSFP28 module and only affects a single data point.

The graphs at the bottom of Figure 4 show the average deviation from the expected packet spacing in parts per million (ppm) for all three devices over time. All graphs show a vertical range of 1 ppm. For all three compared devices, there is a clear correlation between the measured temperature changes and the observed changes of the clock speed error, with an increase in transceiver temperature causing a decrease in clock speed. This matches the expected results of oscillators used for the internal timers running slower for higher temperatures [9], therefore, reporting lower timestamp differences for the same inter-packet spacing. The results also show that changes in temperature, as reported by the transceivers, affect the measured clockspeed deviation differently for each device. Using linear interpolation with the data presented in Figure 4, we determine the effect of temperature changes on the clockspeed as −0.264 ppm/°K for the switch, −0.108 ppm/°K for the NIC, and −0.078 ppm/°K for the capture card.

These measurement results also provide insights into the absolute deviations of clockspeeds from the expected values. The timestamping clocks of the devices are not synchronized to any external reference. Therefore, the accuracy of the timer speed depends on the accuracy of the integrated oscillators. For the three tested devices, the timer deviates from the expected clockspeed by an average of 44.00 ppm for the switch, −3.43 ppm for the NIC, and 22.44 ppm for the capture card.

Additionally, the previously described anomalies in the temperature measurements caused by external factors are also clearly visible in the clock speed error measurements as corresponding peaks.

IV. Discussion and Conclusion

Overall, we showed that temperature has measurable effects on the timestamping functionality of commodity hardware for 100G networks. While there are significant variations due to uncalibrated and unsynchronized internal timers, deviations caused by temperature differences only have negligible effects for the use case of latency measurements using hardware timestamps in comparison. For example, when measuring a delay of 1 ms, the largest observed deviation of −0.264 ppm/°K only results in an absolute error of 264 ns for a temperature variation of 1 °K. With software-based packet processing typically causing jitter in the microsecond range [7], the results in this paper indicate that these effects do not need to be considered or compensated for most applications when using the tested hardware for latency measurements.

The results also suggest that none of the tested devices introduce significant long-term timer errors independent of the temperature variations.

Additionally, the presented measurements demonstrate that when averaging over a large number of timestamp samples, it is possible to detect and measure effects in the sub-nanosecond range. Using this information, it might be possible to gain additional information on the environment of the server, such as cooling problems or administrators accessing a server room by just observing variations in measured timestamps for traffic with known patterns (such as line rate traffic).

This work covered the effects of temperature on the timestamping of received packets for 100G networking devices. Further research on the accuracy and precision of timestamping hardware on commodity networking devices, as well as other effects that might influence the accuracy of latency measurements, is needed to be able to isolate and assess possible effects caused by commodity hardware used as measuring equipment.

While this work covered effects spanning multiple hours, short-term effects and jitter of the measured timestamps may have a more significant impact on the accuracy and precision of the resulting latency measurements. This includes jitter due to the timestamp capture process and short-term variation in the timestamping clock speed.

Finally, the presented results revealed the possibility to gain information on the environment where the measurement was executed by only utilizing latency measurement results. The concept of gaining additional side-channel information from high-precision latency measurement data and the resulting security implications could be further explored in the future.

References

[1] P. Emmerich, S. Gallenmüller, D. Raumer, F. Wohlfart, and G. Carle, "Moongen: A scriptable high-speed packet generator," in *Proceedings of the 2015 ACM Internet Measurement Conference, IMC 2015, Tokyo, Japan, October 28-30, 2015*, K. Cho, K. Fukuda, V. S. Pai, and N. Spring, Eds. ACM, 2015, pp. 275–287. [Online]. Available: https://doi.org/10.1145/2815675.2815692

[2] A. Grigorjew, P. Diederich, T. Hoßfeld, and W. Kellerer, "Affordable measurement setups for networking device latency with sub-microsecond accuracy," p. 5, 2022.

[3] A. Grigorjew, L. K. Schumann, P. Diederich, T. Hoßfeld, and W. Kellerer, "Understanding the performance of different packet reception and timestamping methods in linux," p. 5, 2023.

[4] P. G. Kannan, R. Joshi, and M. C. Chan, "Precise time-synchronization in the data-plane using programmable switching asics," in *Proceedings of the 2019 ACM Symposium on SDN Research, SOSR 2019, San Jose, CA, USA, April 3-4, 2019*. ACM, 2019, pp. 8–20. [Online]. Available: https://doi.org/10.1145/3314148.3314353

[5] R. Kundel, F. Siegmund, J. Blendin, A. Rizk, and B. Koldehofe, "P4STA: high performance packet timestamping with programmable packet processors," in *NOMS 2020 - IEEE/IFIP Network Operations and Management Symposium, Budapest, Hungary, April 20-24, 2020*. IEEE, 2020, pp. 1–9. [Online]. Available: https://doi.org/10.1109/NOMS47738.2020.9110290

[6] R. Kundel, F. Siegmund, and B. Koldehofe, "How to measure the speed of light with programmable data plane hardware?" in *2019 ACM/IEEE Symposium on Architectures for Networking and Communications Systems, ANCS 2019, Cambridge, United Kingdom, September 24-25, 2019*. IEEE, 2019, pp. 1–2. [Online]. Available: https://doi.org/10.1109/ANCS.2019.8901871

[7] S. Gallenmüller, F. Wiedner, J. Naab, and G. Carle, "How low can you go? a limbo dance for low-latency network functions," *J. Netw. Syst. Manage.*, vol. 31, no. 1, dec 2022. [Online]. Available: https://doi.org/10.1007/s10922-022-09710-3

[8] S. Gallenmüller, D. Scholz, H. Stubbe, and G. Carle, "The pos framework: a methodology and toolchain for reproducible network experiments," in *CoNEXT '21: The 17th International Conference on emerging Networking EXperiments and Technologies, Virtual Event, Munich, Germany, December 7 - 10, 2021*, G. Carle and J. Ott, Eds. ACM, 2021, pp. 259–266. [Online]. Available: https://doi.org/10.1145/3485983.3494841

[9] H. Zhou, C. Nicholls, T. Kunz, and H. Schwartz, "Frequency accuracy & stability dependencies of crystal oscillators," *Carleton University, Systems and Computer Engineering, Technical Report SCE-08-12*, 2008.