

Precise Measurement of One-Way Delay and Analysis of Synchronization Issues

Sven Ubik and Vladimir Smotlacha, CESNET, Prague, Czech Republic
{ubik, vs}@cesnet.cz

I. INTRODUCTION

When implementing networks with QoS guarantees, precise measurement of all network QoS characteristics is necessary. We need to observe behaviour of various traffic handling mechanisms on different patterns of data, for example, to see influence of handling a traffic aggregate on a microflow behaviour, which is the case in diffserv networks and to check if the agreed Service Level Specification (SLS) is adhered to by both the user and the network.

We need a tool that can generate a set of data streams of a specified shape and rate, capture these streams as they pass through a network and computes experienced QoS characteristics. To verify behaviour of scheduling mechanisms, precision around 4 microseconds is needed. This is the time to submit one minimum-length packet on a Gigabit Ethernet network.

II. EXISTING TOOLS AND OUR REQUIREMENTS

Commercial network analyzers are expensive and often provide sending and receiving ports on the same device, avoiding the need for time synchronization, but prohibiting to make measurements between two physically separate locations.

Freely available tools usually only measure average throughput [1] or round-trip delay. Detailed insight into instantaneous values of all QoS characteristics measured with selected time granularity is not possible.

We set the following requirements on our measurement system:

- Measurement of packet loss, throughput, delay, delay variation and distribution of delay and delay variation
- Characteristics computed with arbitrarily specified time granularity
- Precision better than 10 microseconds
- Open to further processing of results
- Inexpensive solution

III. CHOICE OF TIME SYNCHRONIZATION

Precise time synchronization is essential for one-way delay measurement. There are three possible ways to adjust local clock. First, a standalone precise clock, such as atomic clock, can be installed in each location. The result is precisely synchronized and accurately set clocks, but at a high cost. Second, each clock can be adjusted using PPS signal from a GPS receiver. Finally, all clocks can be synchronized over a network using NTP protocol. Available estimates of achievable precision using NTP protocol range from 1 to 5 ms [2]. Therefore, we decided to use a GPS receiver in the first generation of our tool. There are certain issues which must be considered in practical use of GPS receivers for time synchronization.

Most GPS receivers provide PPS (Pulse Per Second) signal which can be used to synchronize time in other devices. The absolute accuracy of PPS signal ranges from tens of nanoseconds to 1 microsecond. If measurement is performed over a local network, we can use only one GPS receiver and distribute its PPS signal to all PCs that need to be synchronized. We constructed a voltage converter from TTL to RS-232 and a distribution unit that allows to deploy PPS signal over standard Category 5 cabling.

Another aspect to consider is the resolution of a system clock in a PC, which uses a quartz oscillator 14.318 MHz, divided by 12 to 1.193 MHz, limiting resolution to 0.838 microsecond. Nanokerenel [5] uses the Pentium TSC (Time Stamp Counter) register, which counts CPU ticks, allowing to read local time with one nanosecond resolution on current Pentium processors.

IV. ANALYSIS OF CLOCK SYNCHRONIZATION

We can identify the following factors that influence the computer clock precision:

1. Accuracy of PPS from a GPS receiver - from 50 nanoseconds to 1 microsecond
2. Delay in voltage level converter - approx. 100 ns
3. Delay in cables - 1 microsecond per 300 m given by light speed
4. Delay in the computer caused by processing PPS (interrupt latency)
5. Instability of a quartz oscillator, which includes temperature dependency, long time instability (both partially compensated by PLL/FLL) and short time fluctuations.

When all PCs use the same source of PPS, factors 1 and 2 are eliminated. Factor 3 is small, constant and is eliminated when all PCs get PPS signal over a path of a comparable length. The issue is the variation of factors 4 and 5, which we discuss later.

V. SYSTEM ARCHITECTURE

The system architecture is depicted in Fig. 1. We used RUDE/CRUDE [3] for packet generation and capture. We developed a tool called qosplot which reads packet snapshots produced by CRUDE, computes all QoS characteristics and produces data and command files for gnuplot [6], which can be then used to draw diagrams depicting the characteristics. qosplot can shift specified streams in time, change amplitude (to convert throughput between different network layers) and compute the characteristics with arbitrarily specified time granularity. Produced data file can be used by other applications for further processing of measurement results. We used our system for many measurements performed as a part of the QoS in IP project [4] in CESNET. An example diagram of one-way delay obtained from qosplot is shown in Fig. 2.

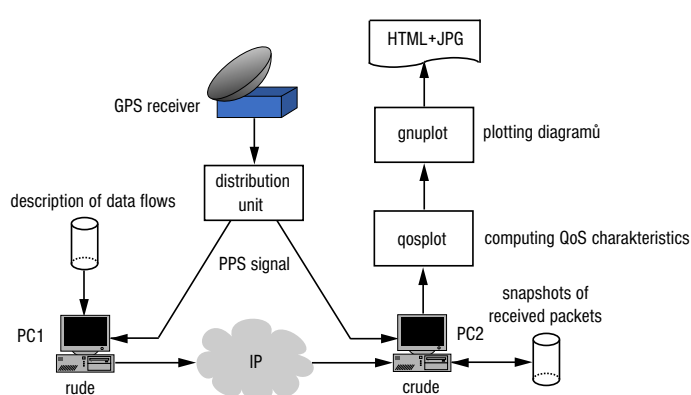


Fig. 1. System architecture

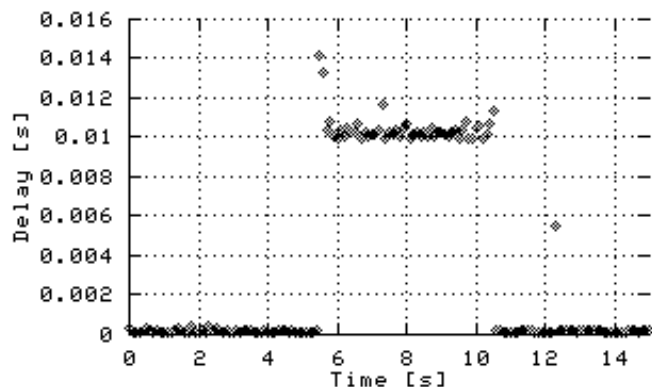


Fig. 2. One-way delay on Gigabit Ethernet with MDRR

VI. PRACTICAL EXPERIENCE

We used two PCs with nanokernel-patched Linux kernels directly connected over a Gigabit Ethernet. Using a PPS software echo in the kernel (an interrupt routine triggered by PPS copies this signal to an output line) and measuring the offset of both signals by a universal counter, we obtained the following PPS processing delay on a lightly-loaded PC:

	mean delay	root Allan variance	standard dev.
P150	11.02	0.83	0.95
P860	8.31	0.34	0.36

We achieved the accuracy of clock synchronization of two computers 2 microseconds during 1 hour measurement. However, these were identical computers located in the same room and under low load. We need to achieve this or better accuracy independently of the load, temperature and computer type.

On Pentium 150 during higher load (e.g., kernel compilation), the mean delay was about 27 microseconds with jitter up to 10 microseconds.

To resolve a problem of dependency on the load, we designed and have custom-manufactured a specialized PCI card. We also wrote a driver for this card. Its functionality is simple: the active PPS edge starts a counter on the card. Later, value of this counter is read in an interrupt routine and the corresponding delay is subtracted from the current time. The count clock is 20 MHz. Therefore, we can determine the time of the PPS edge with the resolution of 50 ns.

Fig. 3 shows timestamps of PPS both without and with our customer card. The measurement was done on Linux PC

Fig. 3. Influence of computer load on PPS processing delay

(Celeron 1GHz) with a standard free running quartz oscillator.

The remaining issue is the instability of the quartz oscillator, mainly its temperature dependency, which we resolved by replacing a standard quartz oscillator with a specialized ovenized oscillator.

VII. USING NTP FOR TIME SYNCHRONIZATION

Although the system described above provides high accuracy at relatively low cost, there is still some hassle involved with installing GPS receivers and distributing PPS signal to PCs. Therefore, we decided to focus on using NTP in the next generation of our system. Contrary to results presented in [2], we found that it is possible to achieve very good synchronization using NTP [7]. Retrieving the internal clock by a software pulse generator and measuring the offset using a counter we found mean offset below 200 microseconds over LAN and a small WAN (3 routers, 2 switches, LANE, MPLS)

VIII. CONCLUSION

We recommend the following for NTP servers or measurement PCs disciplined by PPS signal: CPU speed is not essential (older Pentium is good), disk should be UltraDMA or SCSI, using DMA is essential (without DMA delay increased by more than 100 us), avoid large CPU-intensive processes (cache memory should not be overwritten too often).

We plan to extend the qosplot tool with more features and to perform extensive NTP precision measurements on international basis in the proposed project [8].

REFERENCES

- [1] "Netperf: A Network Performance Benchmark", <http://www.netperf.org>.
- [2] Ulrich Windl, David Dalton, Marc Martinec (eds.). "The NTP FAQ and HOWTO", section 5.1.10, 2000, <http://www.eecis.udel.edu/~ntp/ntpfaq/NTP-a-faq.htm>.
- [3] Juha Laine, Sampo Saaristo, Rui Prior. "RUDE & CRUDE: Real-time UDP Data Emitter and Collector", <http://www.atm.tut.fi/rude/>.
- [4] Sven Ubik, Vladimir Smotlacha. "QoS in IP", research project, CESNET, 2001, <http://www.cesnet.cz/english/project/qosip>.
- [5] D.L. Mills and P.-H. Kamp. "The nanokernel", *Proc. Precision Time and Time Interval (PTI) Applications and Planning Meeting*, Reston VA, November 2000.
- [6] Thomas Williams, Colin Kelley. "gnuplot" — plotting program.
- [7] Vladimr Smotlacha. "Measurement of time servers", Cesnet technical report 18/2001.
- [8] Vladimir Smotlacha. "Accuracy of the Network Time Synchronization", project proposal, Cesnet, 2002.