Calibration of Time Stamp Authorities

Vladimír Smotlacha[1], Petr Tyml[2], Jan Čermák[3]

[1] *CESNET z.s.p.o, Zikova 4, 160 00 Praha 6, Czech Republic, vs@cesnet.cz*
[2] *Czech Technical University, Faculty of Electrical Engineering, Technická 2, 160 00 Praha 6, Czech Republic, tymlp@centrum.cz*
[3] *Institute of Photonics and Electronics, Academy of Sciences, Chaberská 57, 181 00 Praha 8, Czech Republic, cermak@ufe.cz*

Abstract - The time stamp service is a network time service providing time stamp to a specific byte sequence sent to a time stamp authority/ appliance (TSA). The accuracy of the provided time stamp shall be specified by the TSA and should be calibrated. This paper discusses methods for calibration of TSA using a calibration computer and taking into account the impact of the network. The methods have been tested on an experimental TSA placed in different distances from the calibration computer and in different networks.

I. Introduction

Nowadays when the Internet time services are used by many people in everyday life and also in an increasing number of technical applications, the time/frequency metrology faces a question as to whether and how these services should be calibrated. Some may think that the transmission of time over the network is part of informatics and therefore it is the informatics and not metrology that should cope with it. But the network time service may be looked at as a time source, i.e. a clock, and as such it should be calibrated using common procedures adopted in time/frequency metrology. Sometimes we can encounter a confusion regarding this issue mostly arising from a lack of greater knowledge of the other's field i.e., informatics of time/frequency metrology and vice versa.

In this paper we will mention a specific network time service that serves to time stamp electronic documents. This service is provided by time stamp appliances /authorities (TSA) whose calibration will be discussed. The time stamp service which is specified in the RFC3161 [1] is a complex system that includes three segments:

- Cryptography - the service is based on issuing signed certificates and TSA is a special type of certification authority.
- Metrology - TSA includes a clock synchronized to UTC and the service provides time information with a specified accuracy.
- Networking - a network (Internet) is an operating platform for time stamp service. Service availability and provided time accuracy are influenced by network parameters (e.g., a delay).

We will describe two computer systems, a calibration computer (CC) and an experimental TSA, both synchronized to UTC, which allow us to study calibration performance and to carry out practical calibrations of TSA. A great part of this paper is devoted to the results of the time transfer from TSA to CC while TSA is operating in different distances from CC and in different networks.

II. Time stamp service

The TimeStamp protocol (TSP) specified in the RFC3161 [1] defines both the communication protocol and the time stamp format. The time stamp must be provided with a resolution of ≤ 1 s, however the RFC3161 assumes any finer granularity down to 1 μs. The TSP is based on a request message sent by a client (Time Stamp Query – TSQ) and a response message (Time Stamp Reply – TSR) sent by the server. The TSQ can contain arbitrary information (e.g., MD5 hash of a file), which is returned back after being signed in the TSR. This way the information is bound with the time stamp. According to the

RFC3161, the TSP can be implemented either on the 3rd network level (TCP) or on the 4th network level (HTTP or SMTP).

Although there exists an open source implementation (OpenTSA [2]), the majority of operating TSAs are commercial proprietary solutions and therefore it is very difficult even for the operator to give an evidence of TSA time service accuracy. In some countries, the law forces each licensed TSA operator to calibrate his TSA system.

Traditionally, describing the “accuracy” of TSA is mostly reduced to the resolution of internal clock rather than to the accuracy (uncertainty) with respect to UTC as requested in time metrology. This paper discuses how to verify the quality of time information provided by TSA and how to calibrate the accuracy (uncertainty) of the issued time stamp.

III. Basic considerations

In classical clock calibration we measure the time difference between the clock under test and the reference clock which approximates UTC. The time scales generated by the clocks are represented by time-tagged pulses which are applied to a time interval counter that measures the time difference between them. Since the time is defined for a specific point, a correction must be made for the delay to the measurement system.
Obviously, the service provider can take responsibility only for the time information provided at a given point (typically the TSA network interface) but he cannot be responsible for the network transfer delay from client. We can distinguish between two types of measurement: a) provider-oriented (calibration of TSA) and b) client-oriented (calibration of TSA from the client site, i.e. including the network transfer delay).
To calibrate the TSA we use a calibration computer (CC) synchronized to UTC via GPS. The CC provides 1 pps representing its internal time scale T(CC). To verify the method we use an experimental TSA also synchronized to UTC via GPS which provides 1 pps representing its time scale T(TSA). The subject of calibration is the time difference $T_Q – T_S$ where T_Q is the time of TSQ and T_S is the issued time stamp. The uncertainty in T_Q is given by the uncertainty of the T(CC) scale and is assumed to be much smaller than that of T_S defined by the time stamp provider. The time difference $T_Q – T_S$ is affected by: a) the delay in TSQ and TSR processing due to client authorization and authentication, response generation, and TSA queues, b) the network transport delay of TSQ.

IV. Calibration computer

The CC has been designed to meet the following requirements: stable clock synchronized to UTC by an external 1 pps signal, effective platform for calibrating software, open source software, and easily transportable box.

- Hardware and operating system

The CC system is based on a PC compatible hardware with i386 family CPU running operating system Linux. A special hardware card (installed in the PCI clot) is used to capture the 1 pps signal. The card reduces the interrupt latency down to 50 ns. The computer can be optionally equipped by a stable (e.g., OCX) oscillator to increase the clock stability.

- Clock

While the CC system clock utilizes a quartz oscillator at CPU frequency, the clock itself is implemented in the operating system and is accessed by the operating system function *gettimeofday()*. The clock synchronization is made by the *ntpd* process (a fundamental part of the ntp package distribution) that processes the 1 pps signal and runs the digital PLL to discipline the quartz oscillator. Depending on the system kernel type, the clock precision (i.e., the unit in which the time is expressed) is either 1 µs or 1 ns.

The *pps_gen* process generates the 1 pps signal which represents T(CC) at the serial port output pin. In order to avoid an interaction between 1 pps input (which invokes CPU interrupt) and 1 pps output, the latter is shifted by 1 ms. A time interval counter can measure the time difference between input and output and thus verify the T(CC) accuracy.

- Time stamp protocol parsing software

The *proc_tsp* process listens to the network traffic and evaluates the T(CC) based time of each TSQ and TSR. The process merges the corresponding TSQ and TSR and extracts the TSA - provided time stamp from TSR.

For each TSP transaction, the CC creates and stores the record R = (ID_{TSA}, T_Q, T_R, T_S) where

ID_{TSA}	- TSA identification,
T_Q	- time of TSQ,
T_R	- time of TSR,
T_S	- time stamp generated by TSA.

V. Experimental TSA

The experimental TSA has been designed and developed to meet the following requirements: stable clock synchronized to UTC, acceptable system performance, and open source software.

- Hardware and operating system

The experimental TSA requires a standard PC compatible hardware with Linux operating system. The computer is equipped with the same 1 pps signal capture card as the CC.

- Clock

The TSA computer clock implements the T(TSA) time the same way as the CC clock. The *gen_pps* process transmits the T(TSA) time in the form of 1 pps signal to the serial port.

- TSA software

The TSAs are usually implemented as a proprietary commercial software and hardware but an OpenTSA project [2] under the GNU license also exists. The OpenTSA package extends the OpenSSL package by the capability of generating the time stamp certificates according to the RFC3161. It includes the client interface in the form of a module embedded into the Apache www server and therefore the http protocol is used for communication between client and server. The system time T(TSA) is read by the function *gettimeofday()* when the time stamp is generated.

VI. Calibration method

The TSA client sends a TSQ request which is intercepted by the CC which evaluates T_Q and saves it together with the TSQ content. The TSR response is again intercepted, parsed and merged with the stored TSQ. The CC knows all records R = (ID_{TSA}, T_Q, T_R, T_S) and can either immediately verify the TSA accuracy or statistically process a set of samples.

The whole system consists of

- TSA – device under test.
- Calibration computer CC.
- TSA client – the process that asks the TSA for the time stamp.
- Source of 1 pps synchronized to UTC.
- Time interval counter verifying synchronization of T(CC) and T(TSA).

So far we have proposed two calibration methods: provider-oriented and client-oriented.

A. Calibration on TSA site

The first - provider-oriented - method is based on calibration on the TSA site which eliminates the network influence. The goal is to verify the uncertainty claimed by the TSA operator at a defined access point – TSA network interface. The calibration setup is shown in Figure 1. The CC is a passive device that does not affect the TSA operation. The CC can listen to TSA communication by replicating the link traffic using a suitable arrangement that depends on the link protocol. Examples are a splitter for optical links or active network element (hub or a properly configured switch) for the Ethernet link.

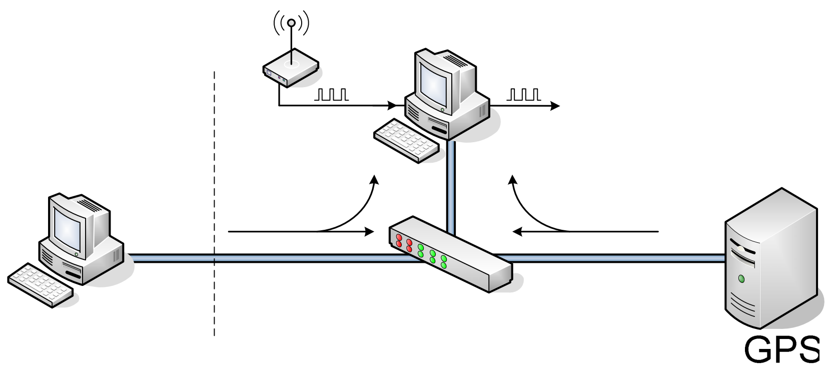

Figure 1: Calibration on the TSA site.

B. Calibration on client site

The second method shown in Figure 2 is client-oriented. It calibrates the TSA service as viewed from the client site and gives an estimation of real system performance at a particular site in network. The results are affected by the network one-way delay between the client site and the TSA site.

As the CC runs also the TSA client process, no network flow replication is necessary. Obviously, this method can also be used in the calibration on the TSA site mentioned above.

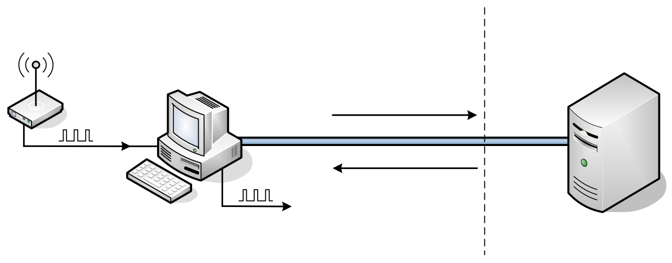

Figure 2: Calibration on the TSA client site.

VII. Implementation

Our CC prototype is based on mini-ITX VIA-EPIA mainbord with a 1 GHz CPU. The 1 pps capture card has been manufactured according to our design. The card driver implements the PPS API which is specified in the RFC2783 [3]. We have installed the Debian 3.1 (sarge) Linux distribution with kernel version 2.4.33. The kernel contains a nanokernel patch which implements the system clock with nanosecond resolution. The clock synchronization is realized by the *ntp* package version 4.2.4. The *gen_pps* and *proc_tsp* programs have been written in C language.

Our experimental TSA prototype is based on an 'off-the-shelf' hardware with the same operating system and 1 pps capture card as the CC. We have installed the *OpenTSA* package and set the time stamp accuracy (uncertainty) in the configuration file to 100 μs. The resolution of time stamps provided is 1 μs.

Measuring the 1 pps output signal, we have verified that our computers' clock accuracy is ≤ 50 μs.
As a result of the measurement, we assume that the uncertainty U(CC) of the time scale T(CC) of our CC prototype is 50 μs. The uncertainty of our experimental TSA time scale T(TSA) is the same, however, to conform to the *OpenTSA* declared value, we assume that U(TSA) is 100 μs.

VIII. Results

We consider the time stamp as incorrect when it does not meet the following conditions:

$$T_S + U(TSA) \geq T_Q - U(CC) \quad (1)$$

$$T_S - U(TSA) \leq T_R + U(CC) \quad (2)$$

where U(CC) is the T(CC) uncertainty and U(TSA) is the T(TSA) uncertainty.

We have looked for incorrect time stamps in the samples recorded by our CC and we have also evaluated the total time T between TSQ and TSR:

$$T = T_R - T_Q \quad (3)$$

Although the RFC3161 [1] does not specify any restriction on T, it is assumed on the order of seconds.

In the following results, time is expressed in UTC seconds since 1.1.1970 0:00:00. (i.e., standard time scale of Unix systems).

A. Calibration on TSA site

We have configured the laboratory IP switch to replicate the experimental TSA traffic to the CC. Since the TSA has a 1 pps synchronized clock, we have found no incorrect time stamps – all conform to conditions (1) and (2).

Example of calibration records of the experimental TSA:

```
        TQ                 TS                 TR           TS - TQ   TR - TQ
1182816842.429474  1182816842.430096  1182816842.444590  0.000622  0.015116
1182816961.660680  1182816961.661345  1182816961.674796  0.000665  0.014116
1182817022.317288  1182817022.317896  1182817022.331280  0.000608  0.013992
1182817081.884046  1182817081.884645  1182817081.898037  0.000599  0.013991
1182817202.158505  1182817202.159104  1182817202.172496  0.000599  0.013991
1182817261.706022  1182817261.706585  1182817261.720014  0.000563  0.013992
1182817322.313907  1182817322.314552  1182817322.328024  0.000645  0.014117
1182817381.876039  1182817381.876601  1182817381.890156  0.000562  0.014117
1182817442.776498  1182817442.777101  1182817442.790615  0.000603  0.014117
1182817502.668434  1182817502.669058  1182817502.682425  0.000624  0.013991
1182817562.638198  1182817562.638789  1182817562.652189  0.000591  0.013991
1182817681.709993  1182817681.710519  1182817681.723985  0.000526  0.013992
1182817742.405323  1182817742.405867  1182817742.419315  0.000544  0.013992
1182817802.200814  1182817802.201425  1182817802.214930  0.000611  0.014116
1182817861.888617  1182817861.889209  1182817861.902734  0.000592  0.014117
```

Table 1. shows the mean value, the standard deviation and the maximum value of the differences $T_S - T_Q$ and $T_R - T_Q$.

n	avg(Ts-Tq)	dev(Ts-Tq)	max(Ts-Tq)	avg(Tr-Tq)	dev(Tr-Tq)	max(Tr-Tq)
3776	0,000399	0,000342	0,005847	0,01425	0,001573	0,059215

Table 1: Results with the experimental TSA.

B. Client site calibration - public TSA

We have tested two external TSAs operated by other providers. The TSA denoted Ext_1, with RTT (round-trip time) of about 9 ms, is located in the Czech Republic. The TSA denoted Ext_2 is located in France with RTT of about 35 ms. The declared accuracy of both external TSAs is 1 s, therefore we assume U(TSA) = 1 s.

Example of calibration records of Ext_2:

```
         TQ                   TS                   TR          TS - TQ  TR - TQ
1178957401.480347  1178957402.000000  1178957405.819134   0.520   4.339
1178957701.795916  1178957702.000000  1178957706.101267   0.204   4.305
1178958002.092410  1178958002.000000  1178958006.435853  -0.092   4.343
1178958301.278801  1178958301.000000  1178958305.597051  -0.279   4.318
1178958601.453004  1178958602.000000  1178958605.954443   0.547   4.501
1178958901.858786  1178958902.000000  1178958906.508451   0.141   4.650
1178959201.468239  1178959201.000000  1178959205.663119  -0.468   4.195
1178959501.465427  1178959502.000000  1178959505.826816   0.535   4.361
1178959801.673220  1178959802.000000  1178959806.001219   0.327   4.328
1178960101.898783  1178960102.000000  1178960106.310356   0.101   4.412
1178960402.124928  1178960402.000000  1178960406.396985  -0.125   4.272
1178960701.320514  1178960701.000000  1178960705.618329  -0.321   4.298
1178961001.569634  1178961002.000000  1178961005.923888   0.430   4.354
1178961301.678217  1178961302.000000  1178961306.011339   0.322   4.333
1178961601.892683  1178961602.000000  1178961606.264602   0.107   4.372
```

Table 2. shows the mean value, the standard deviation and the maximum value of the differences $T_S - T_Q$ and $T_R - T_Q$ for both Ext_1 and Ext_2. It gives evidence that Ext_1 provided some incorrect time stamps that do not meet condition (1) - using U(TSA) = 1 s and considering U(CC) negligible, the condition (1) gives the condition $T_S - T_Q \geq -1$ s. Also noticeable is the maximum value of the difference $T_R - T_Q$ of both Ext_1 and Ext_2, which is in the order of minutes. Table 3. shows that 10211 of 19253 Ext_1 samples are incorrect.

TSA	n	avg(Ts-Tq)	dev(Ts-Tq)	max(Ts-Tq)	avg(Tr-Tq)	dev(Tr-Tq)	max(Tr-Tq)
Ext_1	19253	-2,772	4,539	127,321	1,835	2,098	129,308
Ext_2	17101	0,133	0,714	45,889	5,673	4,942	440,453

Table 2: Results with external TSAs.

TSA	n	Ts-Tq < 0	Ts-Tq < -0.5	Ts-Tq < -0.7	Ts-Tq < -1.0	Ts-Tq < -2.0
Ext_1	19253	12294	11056	10699	10211	8283
Ext_2	17101	7269	277	0	0	0

Table 3: Incorrect time stamps.

IX. Conclusions

We have built a prototype of calibration system and tested it with various TSAs. We distinguish between provider-oriented and client-oriented calibrations. So far we have proposed two calibration methods that can be used on the provider site and on the client site. This paper should be considered a first step in the effort to elaborate methods and procedures for calibration of TSAs and similar network timing services. Though the timing accuracy of the current public TSA services is poor, in the future we can expect more stringent requirements for the accuracy and, consequently, for accurate calibrations.

References

[1] Adams, C., Cain, P., Pinkas, D., Zuccherato, R., "Internet X.509 Public Key Infrastructure, Time-Stamp Protocol (TSP)", RFC 3161, August 2001.
[2] Glozik, Z., OpenTSA project web pages, *http://www.opentsa.org*.
[3]. Mogul, J., Mills, D.L., Brittenson, J., Stone, J., Windl, U., "Pulse-Per-Second API for UNIX-like Operating Systems", RFC 2783, March 2000.
[4] Mills, D.L., "Network Time Protocol Specification, Implementation and Analysis", RFC 1305, March 1992.