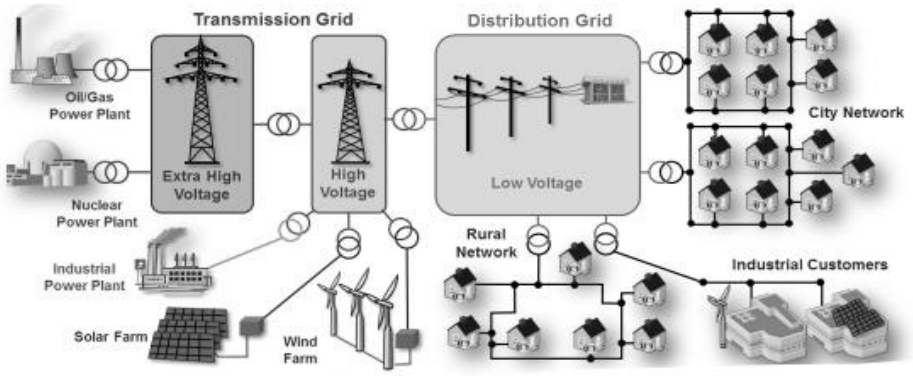
DG ENERGY meeting Brussel/Belgium the 6th of Feb 2016 at 14:30

Tomasz Widomski, in behalf of ELPROMA and DEMETRA H2020 Project Team

Title: Time Gaps Related To Leap Second

DEFINITION

OS – means Operating System, firmware, incl. any software in the chip, that controls sensor, controller, unit or computer (remark: *any modern hardware firmware mostly based on some kind of operating system inside).*



Smart grid infrastructure scheme that is requiring synchronization at each stage

THE LEAP SECOND GAP

A **leap second** is a one-second adjustment that is occasionally applied to Coordinated Universal Time (UTC) in order to keep its time of day close to the mean solar time, and UT1 time. Without such a correction, time reckoned by Earth's rotation drifts away from atomic time (TAI – atomic time scale) because of irregularities in the Earth's rate of rotation. Since this system of correction was implemented in 1972, the 27 leap seconds have been already inserted, the most recent on December 31, 2016 at 23:59:60 UTC (CET 2017 January 1st ,00:59:60) Together with first 10 initial seconds' total amount of leap seconds is now equal 37s. Adding leap second procedure bases on special announcement flag set by decoding special message file from IERS web at: link: "ftp://hpiers.obspm.fr/iers/bul/bulc/bulletinc.52" (please copy this link to your browser to see contends).

The implementation theoretically should give a perfect 61s and the UTC clock effect is [s] should show as follow:

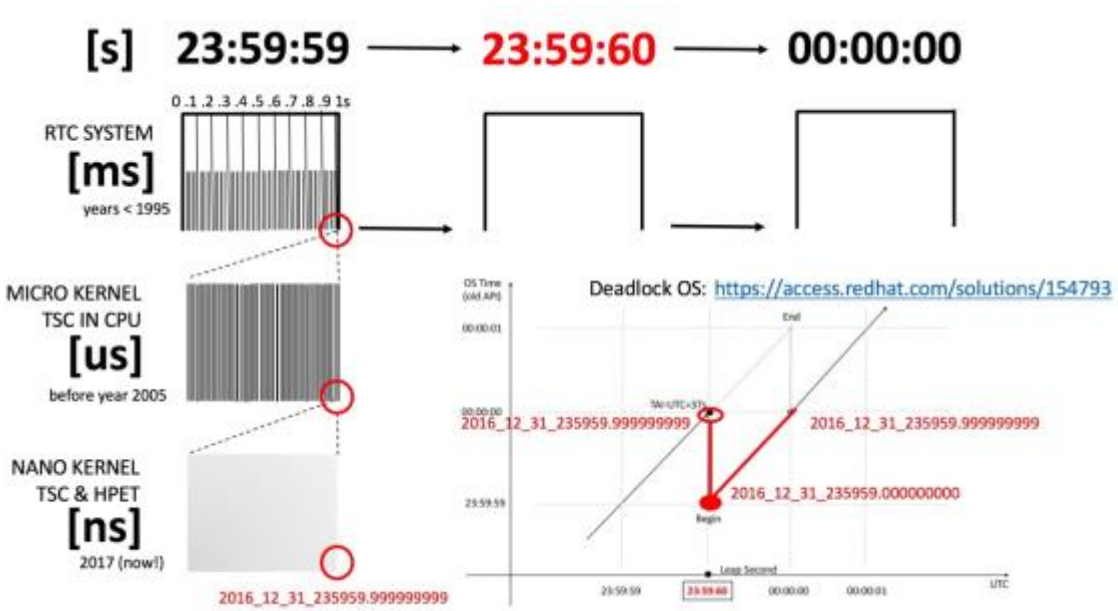


Fig.1 Step Back Clock at 61th second and its impact for duplicating tasks execution inside OS kernel. Tasks scheduled in red top-opened triangle will repeat in execution twice. This could cause kernel panic (see RedHat problem: 154793)

But there are a couple of problems why above structure needs an attention during system deployment. It is true, that computer resolution of time is much below second level. In fact only time displays proceeds time with resolution of 1s. More of modern computers and sensors proceeds time with accuracy of nanoseconds [ns], and some operating system tasks can be scheduled for execution in very small period of fraction of second. According to D. Mills article "*A kernel model for precision timekeeping*", there are possible several side effects of getting time deviations depends on end-user operating system (OS) and its kernel version. We would like to point attention to possible several scenarios of supporting leap second depends on OS version and its kernel (e.g. those implemented in old API generation POSIX Linux system but not only limited to) that might cause problems to any modern computer or industrial sensor. Possible implementations of leap second support at OS kernel are:

1. **Step OS Clock Time Back at the End of the Leap Second**

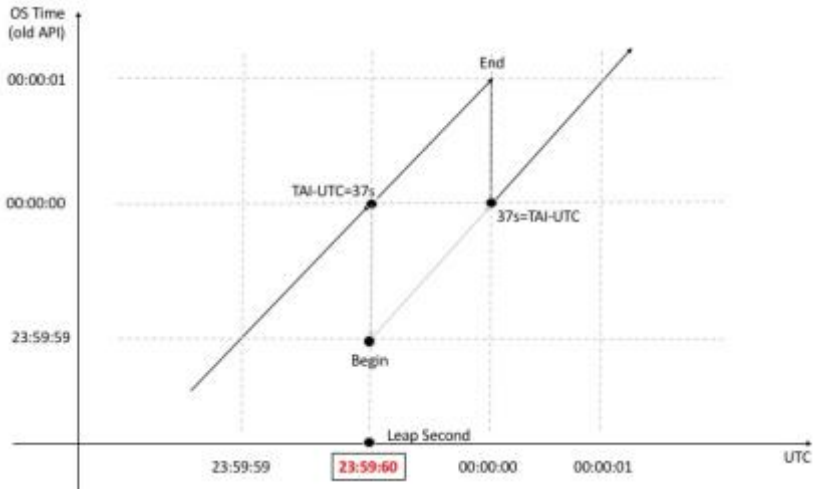


Fig. 2 Leap Second - Step Back Clock at the End of 61th second

In this case time is simply stepped back at the end of an inserted leap second as shown in above figure. Therefore, OS time cannot be monotonic, and thus duplicate time stamps occur after the leap second (e.g. at the beginning of the next UTC day). Mills claims as a result, there can be later time stamps assigned to events which occurred earlier, which can heavily mess up applications using time stamps to order the sequence of events or transactions.

2. **Step OS Clock Time Back at the Beginning of the Leap Second**

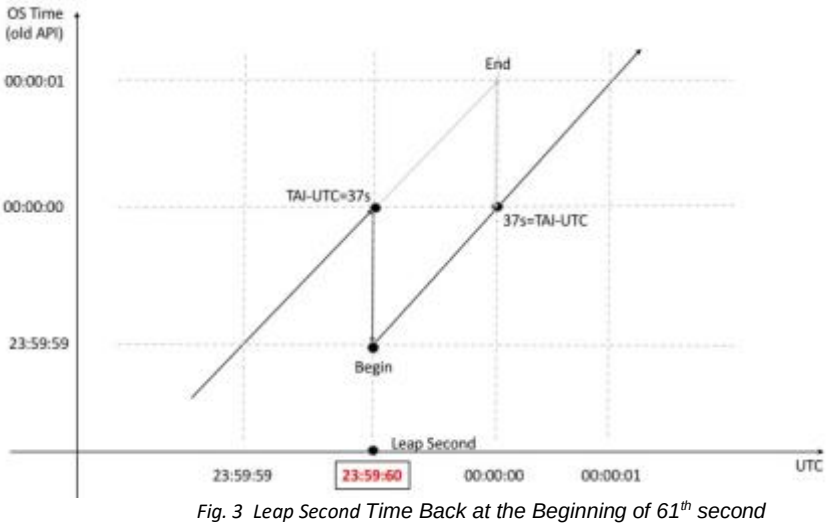


Fig. 3 Leap Second Time Back at the Beginning of 61th second

Time is simply stepped back at the beginning of an inserted leap second as shown above. In this case time is also not monotonic. Mills points the difference from the previous case is that duplicate time stamps occur during the leap second, i.e., at the end of the UTC day. Similarly, there can be later time stamps assigned to events which occurred earlier, which can cause the same confusion as the previous case.

3. Stopping OS Clock Time Counting for Exactly One Second

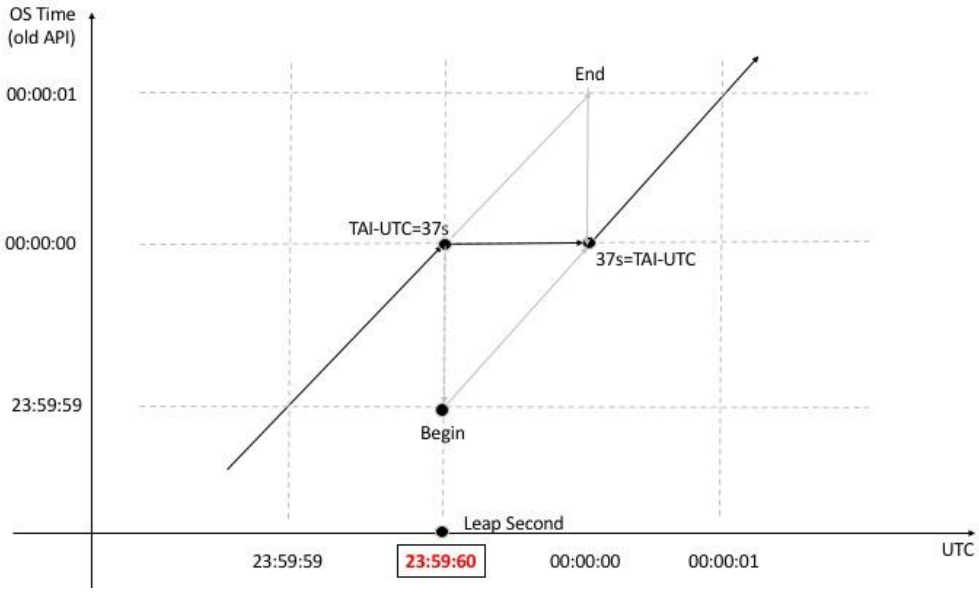


Fig. 4 Leap Second Stopping Clock Time for 1 Second

A modified approach which guarantees strictly monotonic time stamps has been proposed Mills, the inventor of the Network Time Protocol (NTP), who suggested stopping the clock during an inserted leap second, but incrementing the fractions of time stamps by the smallest possible time increment whenever the time is read by an application. This technique gives another the 4th minor scenario (see Fig. 5 below).

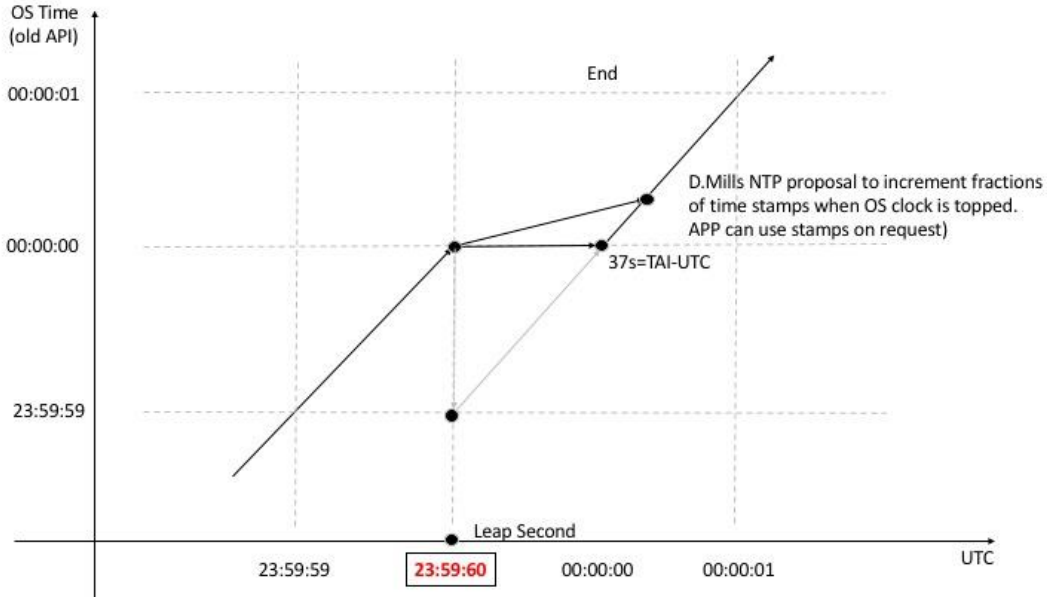


Fig. 5 Dave Mills proposal to Increment Time LSB (the 4th scenario)

Some operating systems like **Microsoft Windows** are not aware of leap second and thus are not prepared to handle it. In such case it may be possible to slew the system time over the leap second. For example, Windows slows the system clock down to half the nominal speed for 2 seconds, so the Windows OS time is again aligned to UTC. This method is not optimal, but at least after the leap second the OS time is correct again.

Nevertheless, there is an important conclusion going out of above discussion. Depends on OS version and its kernel, the leap second can be supported on different ways giving large offset error (counted in seconds), and in some cases it takes hours! **This cause a gap that might volatile security**. Many vendors therefore recommend to not use synchronization a couple of hours before and after leap second (UTC midnight). Besides, if OS clock is just put on hold for one second as shown above and time stamps are all the same during the inserted leap second. OS time does not increase monotonically, and time stamps can't be used to order events due to risk of repeating. Above notes possibly presents a potential gap of today IT synchronization especially important for power girds, telecom, and financial market.

CONCLUSIONS:

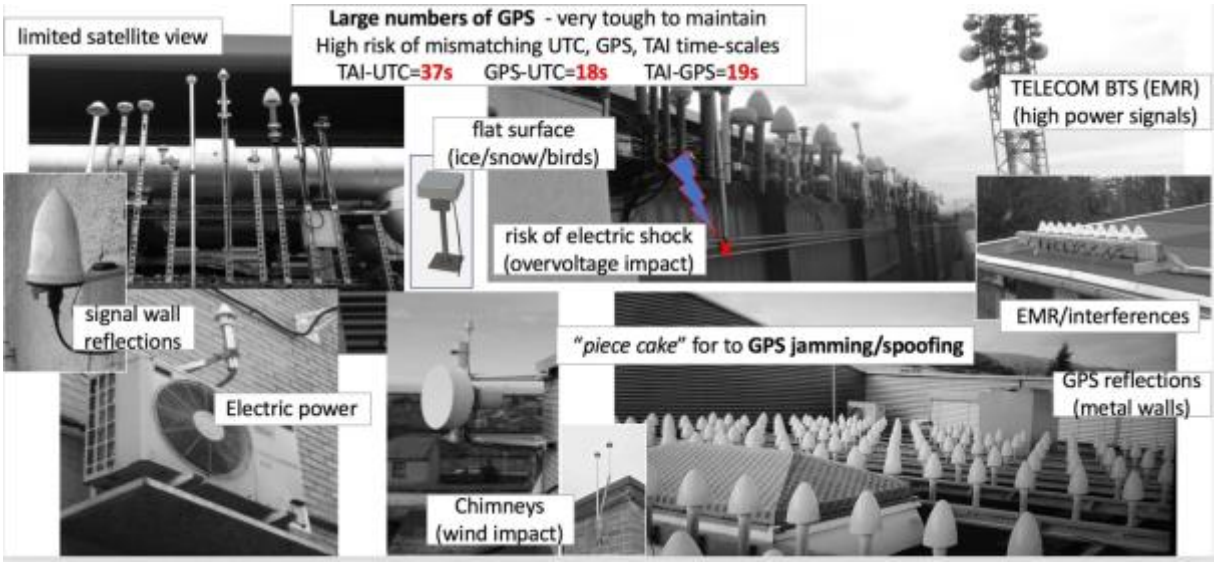
1. Leap second is proceeded locally by each client (slave) individually.
2. Time Servers provides leap second announcement flag to clients (slaves) only.
3. Different OS versions might cause differences in leap second support providing large offsets (gap).
4. Different versions of kernel of OS might cause differences in leap second support too (like above).
5. The leap second announcement flag can be spoofed at receiving GNSS signal level.
6. Leaps second flag cannot be received by GNSS receiver if signal jammed for long term or frequently.
7. The leap second can be false announced at PTP/NTP/IRIG-B level if time servers provide wrong information to slave clients. There is no cryptography protection from IERS for Bulletin C file and therefore file can be corrupted for false leap second other than plane (A) and (B) scenarios.:
   a) *30th of June or/and 31st of December (midnight UTC)*
   b) *31st of March or/and 30th of September (midnight UTC)*
8. The PTP/IEEE1588 (Precision Time Protocol) basis on TAI (atomic time scale) and it includes additional information of volume of *leap_seconds* to be added by PTP-slave. It also supports announcement flag.
9. NTP (Network Time Protocol) is basis on UTC scale and it includes leap second announcement flag too.
10. Some of GNSS receivers can be wrongly set to other than UTC time scales: e.g. TAI or GPS. This might cause large time offsets: 18 (GPS) and 37s (TAI).
11. Single system like GPS might cause errors like one noted 26th of January 2016. Those errors are not possible to detect on simple GNSS receiver level. There are other H2020 projects working on that topic.
12. Jumping time forward/back at client (slave) might cause task duplication, deadlocks and OS kernel panic.

RECOMMENDATIONS:

1. Consider periodical audits of GNSS receivers for leap second announcements. Some services like Galileo will provide authenticated time services that will remind resistant for spoofing (but not for jamming).
2. Verify old GPS (new GNSS) installations. Ensure they follow good practice of installing antennas. Always recommend professional GNSS timing version of receivers (see next page for more details).
3. Installing multiple redundant GNSS receivers located on some long distance to each other, might help reduce risk of using short-range spoofing devices.
4. Consider multilevel redundancy, simultaneously using: (1) GNSS receivers (2) Synchronization via Ethernet (fiber optic PTP/NTP/IRIG-B) (3) local hold over clocks. Work on auditing policy.

APENDIX:

1. Bed practice of installing GPS (GNSS) antennas and receivers



2. Simple way to purchase GPS jammers. Spoofers are available at public Internet too.

3. The six level of components relationship. All used in power distribution industry. Each next level includes previous one or several components (sub-modules). Each level supports: GNSS, PTP/IEEE1588, and local holdover oscillator (local clock).