On Implementing Time

draft-aanchal-time-implementation-guidance-01

A. Malhotra\textsuperscript{1}, K. Teichel\textsuperscript{2}, M. Hoffmann\textsuperscript{3}, W. Toorop\textsuperscript{4}

Boston University\textsuperscript{1}, PTB\textsuperscript{2}, Open Netlabs\textsuperscript{3}, NLnet Labs\textsuperscript{4}
Motivation.

- **functionality and security** of apps hinges on some notion of time.
- choose from **multiple clocks** on systems.
- applications **oblivious to implications** of choosing one or the other clock for implementation
Scope of the Document.

- **Expressing Time**: methods to express time by applications
- **Different clocks**: properties of clocks maintained by digital systems
- **trade-offs** of using one clock over the other
- provides **guidance to help implementers** make an informed choice
Non-Scope of the Document.

- Not specific to architecture of a PC or other devices
- Not specific to any OS.
- Does not deal with how different clocks are available on different PCs or other devices
- No set-in-stone final recommendation.

The final decision would vary depending on the availability of clocks and the security requirements of the specific application under implementation.
• Expressing Time: Absolute vs Relative Time
• Keeping Time: Native Time vs World Time
• Trade-offs of using Native vs World Time
• Current implementation approaches
• POSIX & Windows Example.
Expressing time: Absolute vs Relative time

- **Absolute Time**: expresses an absolute point in time. Nov 6, 2018 12.10pm

  E.g. validity of objects with a limited lifetime that are shared over the network.
Expressing time: Absolute vs Relative time

- **Relative Time:** measures the time interval that has passed from a reference point.

  e.g. Time-to-Live values that determine the length of time for which an object is valid or usable.
Different Clocks – Native Clock

- **Native Clock**: system’s own perception of time
  - obtained by:
    - counting cycles of an oscillator
    - using process CPU times or thread CPU timers
  - returns difference in time between two points
Different Clocks – Native Clock (Properties)

- Properties
  - monotonic
  - immune to vulnerabilities from external time sources
  - quality depends on stability of oscillator or CPU timer
  - Clock drift: clock rate may vary from other systems
• **World Clock**: in synch with other systems.
  
  Obtained by:
  
  • manual settings.
  
  • accessing hardware clock provided by the system which itself is set/updated obtained from an external time source.
  
  • via external sources of time such as Network Time Protocol (NTP), Chrony, SNTP, OpenNTP and others.
Different Clocks – World Clock (Properties)

• Properties
  • can be adjusted for clock drift
  • may stay in sync with other systems
  • manual setting -> misconfiguration errors

  H/W clock access
  • is resource intensive
  • quality of the hardware clock may not be very high leading to a large clock drift if solely relying on it.

  • otherwise, external sources opens up to network attacks
How do software implementations deal with relative time?

COMMON APPROACH
relative time ---> absolute time

Absolute time = ? current system time

Updated by external time sources
Other possible implementation approaches & their trade-offs.

To implement absolute time, no other option but the world clock.

To implement relative time, one MAY use native clock.
POSIX & Microsoft Windows API.

- POSIX: `clock_gettime()` may provide native time
- Microsoft Windows:
  - `GetTickCount` returns 32-bit count
  - `GetTickCount64` returns 64-bit count
Way forward for the draft?