


Writing

Software, technology, sysadmin war stories, and more. 

Sunday, January 29, 2023

Determine durations with monotonic clocks if available

Sometimes, on a lazy weekend afternoon, I use apt-get to pull down the source of something and start grepping it for things that are bound to be interesting. This is one of those afternoons, and I found something silly. While looking for uses of `time_t` in bash, I found a variable called "time_since_start". Uh oh.

bash supports a dynamic variable called "SECONDS" (bet you didn't know that - I didn't), and it's documented as "the number of seconds since shell invocation". I'm sorry to say, that's not quite true. You can totally make it go negative since it's based on *wall time*. Just set the system clock back.

```
root@rpi4b:/tmp# systemctl stop chrony
root@rpi4b:/tmp# echo $SECONDS
11
root@rpi4b:/tmp# date -s "2023-01-01 00:00:00Z"
Sat 31 Dec 2022 04:00:00 PM PST
root@rpi4b:/tmp# echo $SECONDS
-2500987
```

That's an extreme demonstration, but backwards-going wall time happens every time we have a leap second. Granted, we're in a long dry spell at the moment, but it'll probably happen again in our lifetimes. The difference there is just one second, but it could break something if someone relies on that value in a shell script.

Or, how about if the machine comes up with a really bad time for some reason (did your hardware people [cheap out](#) on the BOM and leave off the 25 cent real-time clock on the brand new multi-thousand-dollar server?), the shell gets going, and later chrony (or whatever) fixes it? Same deal, only then it might not be a second. It might be much more.

In the case where the machine comes up with a past date and then jumps forward, SECONDS on a still-running shell from before it's fixed will be far bigger than it should be. I'm pretty sure every Raspberry Pi thinks it's time=0 for a few moments when it first comes up because there's no RTC on the board. Run "last reboot" on one to see what I mean.

I should also mention that bash does other similar things to (attempt to) see how much time has passed. Have you ever noticed that it'll sometimes say "you have new mail", for those rare people who actually use old-school mail delivery? It only checks when enough time has elapsed. I imagine a "negative duration" would mean no more checks.

The lesson here is that wall time is not to be used to measure durations. Any time you see someone subtracting wall times (i.e., anything from `time()` or `gettimeofday()`), worry. Measure durations with a monotonic clock if your device has one. The actual values are a black box, but you can subtract one from the other and arrive at a count of how many of their units have elapsed... ish.

Be sure to pay attention to which monotonic clock you use if you have a choice and there's any possibility the machine can go to sleep. "Monotonic time I have

been running" and "monotonic time since I was booted" are two different things on such devices.

Here's today's bonus "smash head here" moment. From the man pages for `clock_gettime` on a typical Linux box:

`CLOCK_MONOTONIC`: "This clock does not count time that the system is suspended."

Here's the same bit on a current (Ventura) Mac:

`CLOCK_MONOTONIC`: "...and will continue to increment while the system is asleep."

Ah yes, portability. The cause of, and solution to, all of life's software issues.

[More writing](#) | [Contact / send feedback](#)