

Re: a proposed callout API

Source: <http://unix.derkeiler.com/Mailing-Lists/FreeBSD/arch/2006-11/msg00097.html>

- *From:* "Poul-Henning Kamp" <phk@xxxxxxxxxxxxxxxx>
 - *Date:* Tue, 28 Nov 2006 23:28:41 +0000
-

In message <200611281631.19224.jhb@xxxxxxxxxxxx>, John Baldwin writes:

John, I would very much welcome your participation on this.

On the absolute vs relative time thing, this gets far nastier once you start to think about it.

As far as I know, nothing in the kernel asks for sleeps until a given wall-clock (UTC) time. Userland on the other hand often does, and almost never should, but lets leave that behind for a moment. [1]

Suspend/resume is a tricky complication here.

Some sleeps and callouts want to sleep on the "while the CPU is concious" timescale, for instance for pushing dirty pages to disk or collecting usage statistics.

Others want to sleep on the absolute (TAI) timescale, such as TCP retransmission and keepalive timeouts. (The indicative internal/external distinction is not safe btw.)

Right now we don't distinguish between the two cases, and my intention was to leave this for a later stage where we could add flag-bits to signal these desires, once an survey of the kernel code had revealed which were the sensible default.

We can of course add the flags as no-ops already now where this is immediately obvious to us.

Part of the idea was to fix places that abused `tsleep(..., 1)`, etc. to figure out a "real" sleep interval.

This is going to be the major pain in the transition, no matter what we do. Pretty much all short sleep and callout durations are bogus because of the traditional rounding(-up) and HZ granularity.

Re: a proposed callout API

Also, my other API change I was going to do was something like this:

```
msleep() -> mtx_sleep()  
msleep_spin() -> sl_sleep() [...]  
rw_sleep(), sx_sleep() [...]
```

I think this sounds eminently sensible, even if we initially do just the crude thing, getting it expressed in the API allows us to improve the implementation later on.

Poul-Henning

[1] OK, couldn't resist:

Much of this trouble comes about because it used to be that only the UTC clock were available, and programs haven't been rewritten to use `CLOCK_MONOTONIC` where they should.

Examples of bogus behaviour:

Named(8) wants to time zones out on the TAI scale not the UTC scale, so it should not be affected by NTPD stepping the clock but only the uptime of the system. Any amount of time the system is suspended should be tolled on the timer.

Xloc