

Re: Testing for daylight savings on a GMT server

Source: <http://unix.derkeiler.com/Newsgroups/comp.unix.solaris/2005-11/0741.html>

From: Gordon Burditt (gordonb.5iykh_at_burditt.org)

Date: 11/09/05

Date: Wed, 09 Nov 2005 03:12:31 -0000

>> *Your problem is NOT that you need to know whether daylight savings
>> time is in effect. Your problem is that you need to convert from
>> one time zone to another. That, of course, requires that you specify
>> the time zone involved (*NOT* GMT). One way to do this is to set
>> the time zone to the local time zone of a customer (no way to do
>> this portably IN C but by POSIX you can set the TZ environment
>> variable), and have it convert the local time specified by the
>> customer to GMT (using mktime()). Repeat daily, weekly, or whatever
>> to determine the deadline in GMT.*

>[...]

>

>*That's why I've long thought that something like rpc.cmsd ought to record*

I'm not familiar with what rpc.cmsd is. The name reminds me a little of a billing system. Anyway, I assume it does a lot of scheduling by user-input times.

>*the local time _and_ time zone that events are scheduled in, so that it
>can expand repeating events itself in terms of the time zone in which they
>were scheduled.*

cron(8) sorta does this, except I think it uses only one local time zone. The manual page on FreeBSD makes some interesting comments about handling daylight savings transitions and events scheduled around the shift. What it doesn't attempt to do is DISPLAY schedules in advance (or at all).

>*Thus, if I schedule a repeating worldwide event in terms
>of my local time, it may shift for DST relative to GMT, and may have both
>my DST shifts and the viewer's possibly different DST shifts applied to
>it before they see it; that's IMO more intuitive behavior than the current,
>which is that it only schedules properly if both clients and rpc.cmsd are
>running in the same time zone.*

I'll agree here. You may not need to store a timezone for EVERY time; it might be sufficient for a user to set his own time zone and what he schedules is in terms of that time zone. It would be

an interesting feature for cron to allow users to set TZ in a crontab file (which is allowed; it affects the local time zone of the started commands and output of any times they display) and have it determine the time zone of the times contained in the crontab (which is not done; they are all taken as the local system time).

*>Of course, they'd still have to stay in
>sync on the rulebase that determined base and DST offsets and DST start
>and end dates and times for every given timezone that they might deal with.*

I think the key here is to evaluate the conversion at the time it's going to happen (or within the same day). The offsets for something years in the future may change (maybe even several times). The offsets for tomorrow are unlikely to unexpectedly change unless either (1) governments do something really, really stupid, or (2) someone sat on his butt for over a year and didn't get updated rule files. Anything in the future **might** change, but anything within a year is **unlikely** to change. This applies reasonably well to different systems that have independently updated rule sets checked every half year or so.

*>Until such a client/server calendar scheduling program does something like
>that, or until everyone everywhere schedules cross-timezone events in a
>single DST-free timezone (like GMT/UTC), there will be unexpected or
>undesirable results.*

People will not schedule things, especially repeating things, in a single DST-free timezone: that totally defeats the purpose of DST. If people didn't write inflexible policies, or worse, cast-in-stone laws, that said that the operating hours are X to Y, but changed them as the seasons changed, DST would be pointless.

Gordon L. Burditt