

user-space hangs – followup

Source: <http://linux.derkeiler.com/Newsgroups/comp.os.linux.development.system/2006-08/msg00118.html>

- *From:* Dan Miller <dan@xxxxxxxxxxxxxxxxxx>
 - *Date:* Mon, 07 Aug 2006 13:52:41 -0500
-

I had previously posted regarding a problem that I'm seeing here. The characteristics of the problem are that all kernel threads and interrupts appear to continue working, but all user-space processes appear to hang. Symptoms of the "hang" are that network traffic which passes THROUGH the box continue to pass, and pings to the box succeed, but any tcp/udp traffic to or from the box will fail during the hang. Another quirk of the situation is that the hang is not permanent, but will resolve itself after a period of time (anywhere from 15 minutes to 8 hours).

Gil Hammond (GH) had suggested (on this newsgroup) that it sounded as though all user processes were getting stacked up behind a lock/mutex or other resource in the kernel; I've been trying to pursue this line of reasoning further to research this problem (which is still occurring).

Another engineer that I spoke with elsewhere, observed that it didn't sound like a simple mutex deadlock-type issue, because a deadlock like that should never recover. He observed that it sounded more like a counter that was wrapping or otherwise going out of range. Based on that theory, I've been looking back through my driver again – a good exercise anyway, since I've been working on this driver for eight years now, and there are some parts that I don't remember much anymore!!

One thing I'm finding is that we have code such as the following in many parts of the driver:

```
timeout_jiffies = jiffies + jiffies_from_ms(timeout_msec) ;
while ( (rc = test_some_other_condition(device)) &&
(jiffies < timeout_jiffies) ) {
    schedule();
}
// Did we succeed in getting the MUTEX flag?
if ( rc != 0 ) {
    return -1;
}
return 0;
```

I now note that this is actually vulnerable to counter-wrapping; for example, if `timeout_jiffies` gets set to `(MAX_JIFFIES - 5)`, and `test_some_other_condition()` takes more than 5 jiffies, the counter would

user-space hangs – followup

wrap and we'd wait awhile for the timeout to occur. (mind you, we have an 850Mhz Celeron in our machines, so wrapping should still only take about 8.4 minutes). Anyway, my question at this point is this:

Although the counter could wrap and leave me in this loop for awhile, I'm still calling schedule(), as I should. Now this will allow at least kernel-space tasks to proceed, but what about user-space processes?? Would they end up blocked by this loop?? Or will they still get scheduled??

Dan Miller

-----== Posted via Newsfeeds.Com – Unlimited–Unrestricted–Secure Usenet News==-----
<http://www.newsfeeds.com> The #1 Newsgroup Service in the World! 120,000+ Newsgroups
-----= East and West–Coast Server Farms – Total Privacy via Encryption =-----