Subject: RE: dpt_i2o: cycle with interrupts disabled
Posted by mark_salyzyn on Wed, 04 Oct 2006 13:27:22 GMT
View Forum Message <> Reply to Message

Spins like this exist in all drivers that talk directly to responsive hardware, to sleep is not an appropriate option and affects normal runtime performance for real-time hardware, still need a high priority spin for at least part of the failure. Reducing the CPU utilization may be admirable, but would still require a real-time loop that stages to a sleep, an over complication for a rare and fatal condition.

The condition is fatal, this is a Hardware FIFO that responds in less than a PCI cycle and provides an atomic value. Delays in values usually are a result of a hardware bug in the original i960 messaging hardware in early hardware where a multiple read was required to pick up the value from the FIFO, spin was added to deal with this early hardware. Original code spins without timeout, usually at interrupt context, spinning with a timeout is but to add some grace for what should be an impossible condition. Failure requires Firmware to go 'nutz', hardware failures on the card, overheating or power supply levels to PCI slot being inappropriate.

Since the Linux community takes no more patches for this driver except for a bona-fide bug, ie a failure that can be averted, I do not expect any code space mitigation for this issue to be accepted.

Sincerely -- Mark Salyzyn

> -----Original Message-----
> From: Vasily Averin [mailto:vvs@sw.ru]
> Sent: Wednesday, October 04, 2006 5:16 AM
> To: Salyzyn, Mark
> Cc: devel@openvz.org
> Subject: Re: dpt_i2o: cycle with interrupts disabled
>
>
> Hello Mark,
>
> Of course I cannot exclude some hardware issues. However I
> would note that your
> driver is incorrect.
> First of all I would note that NMI watchdog have 5 second
> timeout and it detects
> busy loop in your driver correctly. Ok, we can make timeout
> in your driver
> lesser that 5 seconds, but it seems for me it is a bad idea
> to have such loops
> at all. If you have found that hardware is not ready, you can

> return error to
> scsi midlayer. If you want to wait sometime, you can free the
> locks, enable
> interrupts, go to sleep and free the CPU for the other processes.
>
> thank you,
> Vasily Averin
>
> SWsoft Virtuozzo/OpenVZ Linux kernel team
>
> Salyzyn, Mark wrote:
> > This is a sign of a serious hardware problem. The timeout
> of this loop
> > was set to 30 seconds when the NMI watchdog used to be set to 60
> > seconds. Now that it is 30 seconds, I recommend you drop the loop
> > timeout to 20 seconds so that it times out before the
> system notices.
> >
> > The serious hardware problem will then turn from a panic
> into a slightly
> > more graceful failure to talk to the adapter as it is no longer
> > responsive and the devices will all go offline. I suggest
> you look into
> > why the Adapter is failing on your system; look into Power
> Supply, PCI
> > bridge, Motherboard or the Card itself. I have no body of
> experience as
> > to why you may see this failure, but you may wish to contact Adaptec
> > Technical support as they may have some sage advice.
> >
> > Sincerely -- Mark Salyzyn
> >
> >
> >> -----Original Message-----
> >> From: Vasily Averin [mailto:vvs@sw.ru]
> >> Sent: Tuesday, October 03, 2006 5:29 AM
> >> To: Salyzyn, Mark
> >> Cc: devel@openvz.org
> >> Subject: dpt_i2o: cycle with interrupts disabled
> >>
> >>
> >> Mark,
> >>
> >> I would like to tell you that we have included your driver
> >> into our kernels.
> >> Unfortunately it does not work well and our customers who
> >> tried to use it
> >> instead of i2o_block driver claims on the node lockups.

> >> We have received error messages, it shows that NMI watchdog
> >> detected that your
> >> driver loops in the following cycle up to 30 sec with
> >> interrupts disabled:
> >>
> >> scsi_dispatch_cmd()   (spin_lock_irqsave(host->host_lock, flags);)
> >> host->hostt->queuecommand() == adpt_queue()
> >>   adpt_scsi_to_i2o()
> >>   adpt_i2o_post_this():
> >> ...
> >>   ulong timeout = jiffies + 30*HZ;
> >>   do {
> >>   rmb();
> >>   m = readl(pHba->post_port);
> >>   if (m != EMPTY_QUEUE) {
> >>   break;
> >>   }
> >>   if(time_after(jiffies,timeout)){
> >>   printk(KERN_WARNING"dpti%d: Timeout
> >> waiting for message
> >> frame!\n", pHba->unit);
> >>   return -ETIMEDOUT;
> >>   }
> >>   } while(m == EMPTY_QUEUE);
> >> ...
> >>
> >> Have you probably some ideas how to fix this issue in a proper way?
> >>
> >> Thank you,
> >> Vasily Averin
> >>
> >> SWsoft Virtuozzo/OpenVZ Linux kernel team
> >>
> >
> >
>
>