Re: [CFT][PATCH] new scheduler policy

Source: http://linux.derkeiler.com/Mailing−Lists/Kernel/2003−08/6688.html

---

From: Haoqiang Zheng (*hzheng_at_cs.columbia.edu*)
Date: 08/25/03

```
Date:   Mon, 25 Aug 2003 09:47:50 −0400
To: William Lee Irwin III <wli@holomorphy.com>
```

William Lee Irwin III wrote:

>On Tue, Aug 19, 2003 at 12:24:17PM +0200, Mike Galbraith wrote:
>
>
>>Test−starve.c starvation is back (curable via other means), but irman2 is
>>utterly harmless. Responsiveness under load is very nice until I get to
>>the "very hefty" end of the spectrum (expected). Throughput is down a bit
>>at make −j30, and there are many cc1's running at very high priority once
>>swap becomes moderately busy. OTOH, concurrency for the make −jN in
>>general appears to be up a bit. X is pretty choppy when moving windows
>>around, but that _appears_ to be the newer/tamer backboost bleeding a
>>kdeinit thread a bit too dry. (I think it'll be easy to correct, will let
>>you know if what I have in mind to test that theory works out). Ending on
>>a decidedly positive note, I can no longer reproduce priority inversion
>>troubles with xmms's gl thread, nor with blender.
>>(/me wonders what the reports from wine/game folks will be like)
>>
>>
>
>Someone else appears to have done some work on the X priority inversion
>issue who I'd like to drag into this discussion, though there doesn't
>really appear to be an opportune time.
>
>Haoqiang, any chance you could describe your solutions to the X priority
>inversion issue?
>
>
>−− wli
>
>
I didn't follow the whole discussion. But from what wli has described to me, the problem (xmms skips frames) is pretty like a X scheduler problem.

X server works like this:
"The X server uses select(2) to detect clients with pending input. Once the set of clients with pending input is determined, the X server starts

executing requests from the client with the smallest file descriptor.
Each client has a buffer which is used to read some data from the
network connection, that buffer can be resized to hold unusually large
requests, but is typically 4KB. Requests are executed from each client
until either the buffer is exhausted of complete requests or after ten
requests. After requests are read from all of the ready clients, the
server determines whether any clients still have complete requests in
their buffers. If so, the server foregoes the select(2) call and goes
back to processing requests for those clients. When all client input
buffers are exhausted of complete requests, the X server returns to
select(2) to await additional data. "
−−− Keith Packard, "Efficiently Scheduling {X} Clients", FREENIX−00,

Basically, the X server does a round robin for all the clients with
pending input. It's not surprising that xmms skip frames when there are
a lot of "heavy" x requests pending. I am not sure if this the cause of
the problem that you guys are talking about. But anyway, if this the
cause, here is my 2 cents:

I think the scheduler of X server has to be "smarter". It has to know
which X client is more "important" and give the important client a high
priority, otherwise the priority inversion problem will be
un−avoidable. Suppose the system can provide something like
"get_most_important_client()" , the X server can be fixed this way:
The X server calls get_most_important_client() before it starts to
handle an X request. If the return is not NULL, it handles the request
from this "important" client. This way, an "important" x client only
need to wait a maximun of a single X request (instead of unlimited
number of X requests) to get served.

The problem now is how can we decide which X client is the most
important? Well, I guess there are a lot of solutions. I have a kernel
based solution to this question. The basic idea is: keep the
processes blocked by X server in the runqueue. If a certain process (P)
of this kind is scheduled, the kernel switch to the X server instead. If
the X server get scheduled in this way, it can handle the X requests
from this very process (P). If you have interest, you can take a look
at http://www.ncl.cs.columbia.edu/publications/cucs−005−03.pdf .

Let me know your comments...

−
To unsubscribe from this list: send the line "unsubscribe linux−kernel" in
the body of a message to majordomo@vger.kernel.org
More majordomo info at http://vger.kernel.org/majordomo−info.html
Please read the FAQ at http://www.tux.org/lkml/