Subject: Re: [PATCH v3 08/10] IPC: message queue copy feature introduced
Posted by Stanislav Kinsbursky on Sat, 11 Aug 2012 13:31:53 GMT
View Forum Message <> Reply to Message

> Hi Stanislav,
>
> 2012/8/10 Stanislav Kinsbursky <skinsbursky@parallels.com>:
>> This patch is required for checkpoint/restore in userspace.
>> IOW, c/r requires some way to get all pending IPC messages without deleting
>> them from the queue (checkpoint can fail and in this case tasks will be resumed,
>> so queue have to be valid).
>> To achive this, new operation flag MSG_COPY for sys_msgrcv() system call was
>> introduced. Also, copy counter was added to msg_queue structure. It's set to
>> zero by default and increases by one on each copy operation and decreased by
>> one on each receive operation until reaches zero.
> Is msq->q_copy_cnt really necessary?
> As far as I can see user space needs the ability to read the n-th message.
>
> The implementation adds a state variable to the kernel, adds two
> automatic updates of the state into msgrcv() (an increase during
> MSG_COPY, a decrease during normal receives) and adds a msgctl() to
> set the state to a certain value.
>
> a) What about the simpler approach:
> - if MSG_COPY is set, then @mtype is interpreted as the number of the
> message that should be copied.
> If there are less than @mtype messages, then -ENOMSG is returned.

Hi, Manfred.
Your approach is simplier, but makes the call less generic and adds
limitations.
I.e. sys_msgrcv() allows you to receive message by type. And from my pow
this logic have to be preserved - you can specify type and then copy all
the messages of specified type.

> b) I do not understand the purpose of the decrease of msq->q_copy_cnt:
> Do you want to handle normal msgrcv() calls in parallel with
> msgrcv(|MSG_COPY) calls?

Actually, I'm not going to copy a message from a queue, when somebody is
reading from it. But better to handle this case by decreasing
msq->q_copy_cnt, because otherwise this counter becomes invalid in case
of somebody is reading from queue. And this logic is similar to new
"peek" logic for sockets (introduced in 3.4 or 3.5).
But I understand, that in case of queue with messages with different
types this approach works only if mtype is not specified for copy
operation. Otherwise result is unpredictable.

> I don't think that this will work:
> What if msq->q_copy_cnt is 1 and and msgrcv() call receives the 20th
> message in the queue?

By "receives" you mean "copied"? If so, then it can happen only if mtype
was specified. And this logic is a part of current implementation.

>
> --
> Manfred
