
Subject: Re: [PATCH v8 4/5] ipc: message queue copy feature introduced

Posted by [akpm](#) on Wed, 24 Oct 2012 21:41:28 GMT

[View Forum Message](#) <> [Reply to Message](#)

On Wed, 24 Oct 2012 19:35:20 +0400

Stanislav Kinsbursky <skinsbursky@parallels.com> wrote:

```
> 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. If this flag was specified, then mtype is interpreted as number of
> the message to copy.
> If MSG_COPY is set, then kernel will allocate dummy message with passed size,
> and then use new copy_msg() helper function to copy desired message (instead of
> unlinking it from the queue).
>
> ...
>
> @@ -777,19 +777,48 @@ long do_msgrcv(int msqid, void __user *buf, size_t bufsz, long
msgtyp,
> struct msg_msg *msg;
> int mode;
> struct ipc_namespace *ns;
> +#ifdef CONFIG_CHECKPOINT_RESTORE
> + struct msg_msg *copy = NULL;
> + unsigned long copy_number = 0;
> +#endif
>
> if (msqid < 0 || (long) bufsz < 0)
> return -EINVAL;
> + if (msgflg & MSG_COPY) {
> +#ifdef CONFIG_CHECKPOINT_RESTORE
> +
> + if (msgflg & MSG_COPY) {
```

This test isn't needed.

```
> + copy_number = msgtyp;
> + msgtyp = 0;
> + }
> +
> + /*
> + * Create dummy message to copy real message to.
> + */
> + copy = load_msg(buf, bufsz);
> + if (IS_ERR(copy))
```

```

> + return PTR_ERR(copy);
> + copy->m_ts = bufsz;
> + #else
> + return -ENOSYS;
> + #endif
> + }
> mode = convert_mode(&msgtyp, msgflg);
> ns = current->nsproxy->ipc_ns;
>
> msq = msg_lock_check(ns, msqid);
> - if (IS_ERR(msq))
> + if (IS_ERR(msq)) {
> + #ifdef CONFIG_CHECKPOINT_RESTORE
> + if (msgflg & MSG_COPY)
> + free_msg(copy);
> + #endif
> return PTR_ERR(msq);
> + }
>
> for (;;) {
> struct msg_receiver msr_d;
> struct list_head *tmp;
> + long msg_counter = 0;
>
> msg = ERR_PTR(-EACCES);
> if (ipcperms(ns, &msq->q_perm, S_IRUGO))
> @@ -809,8 +838,16 @@ long do_msgrcv(int msqid, void __user *buf, size_t bufsz, long
msgtyp,
> if (mode == SEARCH_LESSEQUAL &&
> walk_msg->m_type != 1) {
> msgtyp = walk_msg->m_type - 1;
> + #ifdef CONFIG_CHECKPOINT_RESTORE
> + } else if (msgflg & MSG_COPY) {
> + if (copy_number == msg_counter) {
> + msg = copy_msg(walk_msg, copy);
> + break;
> + }
> + #endif
> } else
> break;
> + msg_counter++;
> }
> tmp = tmp->next;
> }
> @@ -823,6 +860,10 @@ long do_msgrcv(int msqid, void __user *buf, size_t bufsz, long
msgtyp,
> msg = ERR_PTR(-E2BIG);
> goto out_unlock;

```

```

> }
> + #ifdef CONFIG_CHECKPOINT_RESTORE
> + if (msgflg & MSG_COPY)
> + goto out_unlock;
> + #endif
> list_del(&msg->m_list);
> msq->q_qnum--;
> msq->q_rtime = get_seconds();
> @@ -906,8 +947,13 @@ out_unlock:
> break;
> }
> }
> - if (IS_ERR(msg))
> + if (IS_ERR(msg)) {
> + #ifdef CONFIG_CHECKPOINT_RESTORE
> + if (msgflg & MSG_COPY)
> + free_msg(copy);
> + #endif
> return PTR_ERR(msg);
> + }
>
> bufsz = msg_handler(buf, msg, bufsz);
> free_msg(msg);

```

It's all a bit ugly, but I don't really see much we can practically do about that.

You could add something like

```

#ifdef CONFIG_CHECKPOINT_RESTORE
static inline void free_copy(void *p, int msgflg, struct msg_msg *copy)
{
    if (IS_ERR(p) && (msgflg & MSG_COPY))
        free_msg(copy);
}
#else
/* As a macro because `copy' will be undefined */
#define free_copy(p, msgflg, copy) do {} while (0)
#endif

```

and use that in a couple of places. But that won't help much.
