Subject: [PATCH v7 08/10] IPC: message queue receive cleanup
Posted by Stanislav Kinsbursky on Thu, 18 Oct 2012 10:23:17 GMT
View Forum Message <> Reply to Message

This patch moves all message related manipulation into one function msg_fill().
Actually, two functions because of the compat one.

Signed-off-by: Stanislav Kinsbursky <skinsbursky@parallels.com>

```
---
 include/linux/msg.h |    5 +++--
 ipc/compat.c        |   45 ++++++++++++++++++++-------------------------
 ipc/msg.c           |   44 ++++++++++++++++++++++++---------------------
 3 files changed, 45 insertions(+), 49 deletions(-)

diff --git a/include/linux/msg.h b/include/linux/msg.h
index 7a4b9e9..f38edba 100644
--- a/include/linux/msg.h
+++ b/include/linux/msg.h
@@ -34,7 +34,8 @@ struct msg_queue {
 /* Helper routines for sys_msgsnd and sys_msgrcv */
 extern long do_msgsnd(int msqid, long mtype, void __user *mtext,
    size_t msgsz, int msgflg);
-extern long do_msgrcv(int msqid, long *pmtype, void __user *mtext,
-   size_t msgsz, long msgtyp, int msgflg);
+extern long do_msgrcv(int msqid, void __user *buf, size_t bufsz, long msgtyp,
+       int msgflg,
+       long (*msg_fill)(void __user *, struct msg_msg *, size_t ));

 #endif /* _LINUX_MSG_H */
diff --git a/ipc/compat.c b/ipc/compat.c
index 84d8efd..1fd8d1c 100644
--- a/ipc/compat.c
+++ b/ipc/compat.c
@@ -313,6 +313,20 @@ static long do_compat_semctl(int first, int second, int third, u32 pad)
  return err;
 }

+long compat_do_msg_fill(void __user *dest, struct msg_msg *msg, size_t bufsz)
+{
+ struct compat_msgbuf __user *msgp = dest;
+ size_t msgsz;
+
+ if (put_user(msg->m_type, &msgp->mtype))
+  return -EFAULT;
+
+ msgsz = (bufsz > msg->m_ts) ? msg->m_ts : bufsz;
+ if (store_msg(msgp->mtext, msg, msgsz))
+  return -EFAULT;
```

```
+ return msgsz;
+}
+
 #ifdef CONFIG_ARCH_WANT_OLD_COMPAT_IPC
 long compat_sys_semctl(int first, int second, int third, void __user *uptr)
 {
@@ -344,10 +358,6 @@ long compat_sys_msgsnd(int first, int second, int third, void __user
*uptr)
 long compat_sys_msgrcv(int first, int second, int msgtyp, int third,
       int version, void __user *uptr)
 {
- struct compat_msgbuf __user *up;
- long type;
- int err;
-
  if (first < 0)
   return -EINVAL;
  if (second < 0)
@@ -355,23 +365,14 @@ long compat_sys_msgrcv(int first, int second, int msgtyp, int third,

  if (!version) {
   struct compat_ipc_kludge ipck;
-  err = -EINVAL;
   if (!uptr)
-   goto out;
-  err = -EFAULT;
+   return -EINVAL;
   if (copy_from_user (&ipck, uptr, sizeof(ipck)))
-   goto out;
+   return -EFAULT;
   uptr = compat_ptr(ipck.msgp);
   msgtyp = ipck.msgtyp;
  }
- up = uptr;
- err = do_msgrcv(first, &type, up->mtext, second, msgtyp, third);
- if (err < 0)
-  goto out;
- if (put_user(type, &up->mtype))
-  err = -EFAULT;
-out:
- return err;
+ return do_msgrcv(first, uptr, second, msgtyp, third, compat_do_msg_fill);
 }
 #else
 long compat_sys_semctl(int semid, int semnum, int cmd, int arg)
@@ -392,16 +393,8 @@ long compat_sys_msgsnd(int msqid, struct compat_msgbuf __user
*msgp,
 long compat_sys_msgrcv(int msqid, struct compat_msgbuf __user *msgp,
```

```
         compat_ssize_t msgsz, long msgtyp, int msgflg)
 {
- long err, mtype;
-
- err =  do_msgrcv(msqid, &mtype, msgp->mtext, (ssize_t)msgsz, msgtyp, msgflg);
- if (err < 0)
-  goto out;
-
- if (put_user(mtype, &msgp->mtype))
-  err = -EFAULT;
- out:
- return err;
+ return do_msgrcv(msqid, msgp, (ssize_t)msgsz, msgtyp, msgflg,
+    compat_do_msg_fill);
 }
 #endif

diff --git a/ipc/msg.c b/ipc/msg.c
index 3c13b99..028ab87 100644
--- a/ipc/msg.c
+++ b/ipc/msg.c
@@ -767,15 +767,30 @@ static inline int convert_mode(long *msgtyp, int msgflg)
  return SEARCH_EQUAL;
 }

-long do_msgrcv(int msqid, long *pmtype, void __user *mtext,
-  size_t msgsz, long msgtyp, int msgflg)
+static long do_msg_fill(void __user *dest, struct msg_msg *msg, size_t bufsz)
+{
+ struct msgbuf __user *msgp = dest;
+ size_t msgsz;
+
+ if (put_user(msg->m_type, &msgp->mtype))
+  return -EFAULT;
+
+ msgsz = (bufsz > msg->m_ts) ? msg->m_ts : bufsz;
+ if (store_msg(msgp->mtext, msg, msgsz))
+  return -EFAULT;
+ return msgsz;
+}
+
+long do_msgrcv(int msqid, void __user *buf, size_t bufsz, long msgtyp,
+       int msgflg,
+       long (*msg_handler)(void __user *, struct msg_msg *, size_t ))
 {
  struct msg_queue *msq;
  struct msg_msg *msg;
  int mode;
```

```
  struct ipc_namespace *ns;

- if (msqid < 0 || (long) msgsz < 0)
+ if (msqid < 0 || (long) bufsz < 0)
    return -EINVAL;
  mode = convert_mode(&msgtyp, msgflg);
  ns = current->nsproxy->ipc_ns;
@@ -816,7 +831,7 @@ long do_msgrcv(int msqid, long *pmtype, void __user *mtext,
     * Found a suitable message.
     * Unlink it from the queue.
     */
-   if ((msgsz < msg->m_ts) && !(msgflg & MSG_NOERROR)) {
+   if ((bufsz < msg->m_ts) && !(msgflg & MSG_NOERROR)) {
     msg = ERR_PTR(-E2BIG);
     goto out_unlock;
    }
@@ -843,7 +858,7 @@ long do_msgrcv(int msqid, long *pmtype, void __user *mtext,
   if (msgflg & MSG_NOERROR)
    msr_d.r_maxsize = INT_MAX;
   else
-   msr_d.r_maxsize = msgsz;
+   msr_d.r_maxsize = bufsz;
   msr_d.r_msg = ERR_PTR(-EAGAIN);
   current->state = TASK_INTERRUPTIBLE;
   msg_unlock(msq);
@@ -906,29 +921,16 @@ out_unlock:
  if (IS_ERR(msg))
   return PTR_ERR(msg);

- msgsz = (msgsz > msg->m_ts) ? msg->m_ts : msgsz;
- *pmtype = msg->m_type;
- if (store_msg(mtext, msg, msgsz))
-  msgsz = -EFAULT;
-
+ bufsz = msg_handler(buf, msg, bufsz);
  free_msg(msg);

- return msgsz;
+ return bufsz;
 }

 SYSCALL_DEFINE5(msgrcv, int, msqid, struct msgbuf __user *, msgp, size_t, msgsz,
   long, msgtyp, int, msgflg)
 {
- long err, mtype;
-
- err =  do_msgrcv(msqid, &mtype, msgp->mtext, msgsz, msgtyp, msgflg);
- if (err < 0)
```

```
-  goto out;
-
- if (put_user(mtype, &msgp->mtype))
-  err = -EFAULT;
-out:
- return err;
+ return do_msgrcv(msqid, msgp, msgsz, msgtyp, msgflg, do_msg_fill);
 }

 #ifdef CONFIG_PROC_FS
```