
Subject: [PATCH 3/5] test: IPC message queue copy feature test update
Posted by [Stanislav Kinsbursky](#) on Fri, 26 Oct 2012 11:06:07 GMT
[View Forum Message](#) <> [Reply to Message](#)

This update fixes coding style problems (80-characters line and others).
Also, it fixes test to work with new IPC sysctls (instead of using
experimental API logic, which was thrown away and replaced by sysctls).

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

```
include/linux/msg.h          | 3 +
ipc/compat.c                 | 3 +
ipc/msg.c                   | 2 -
ipc/util.c                   | 2 -
tools/testing/selftests/ipc/msgque.c | 83 ++++++-----
5 files changed, 55 insertions(+), 38 deletions(-)
```

```
diff --git a/include/linux/msg.h b/include/linux/msg.h
```

```
index f38edba..391af8d 100644
```

```
--- a/include/linux/msg.h
```

```
+++ b/include/linux/msg.h
```

```
@ @ -36,6 +36,7 @ @ extern long do_msgsnd(int msqid, long mtype, void __user *mtext,
    size_t msgsz, 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 ));
```

```
+    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 eb3ea16..2547f29 100644
```

```
--- a/ipc/compat.c
```

```
+++ b/ipc/compat.c
```

```
@ @ -365,7 +365,8 @ @ long compat_sys_msgrcv(int first, int second, int msgtyp, int third,
    uptr = compat_ptr(ipck.msgp);
```

```
    msgtyp = ipck.msgtyp;
```

```
}
```

```
- return do_msgrcv(first, uptr, second, msgtyp, third, compat_do_msg_fill);
```

```
+ 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)
```

```
diff --git a/ipc/msg.c b/ipc/msg.c
```

```
index f9774ff..91873df 100644
```

```
--- a/ipc/msg.c
```

```
+++ b/ipc/msg.c
```

```

@@ -771,7 +771,7 @@ static long do_msg_fill(void __user *dest, struct msg_msg *msg, size_t
bufsz)

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 ))
+    long (*msg_handler)(void __user *, struct msg_msg *, size_t))
{
    struct msg_queue *msq;
    struct msg_msg *msg;
diff --git a/ipc/util.c b/ipc/util.c
index a961e46..74e1d9c 100644
--- a/ipc/util.c
+++ b/ipc/util.c
@@ -282,7 +282,7 @@ int ipc_addid(struct ipc_ids* ids, struct kern_ipc_perm* new, int size)

    if (next_id < 0) {
        new->seq = ids->seq++;
-    if(ids->seq > ids->seq_max)
+    if (ids->seq > ids->seq_max)
        ids->seq = 0;
    } else {
        new->seq = ipcid_to_seqx(next_id);
diff --git a/tools/testing/selftests/ipc/msgqueue.c b/tools/testing/selftests/ipc/msgqueue.c
index e2d6a64..d664182 100644
--- a/tools/testing/selftests/ipc/msgqueue.c
+++ b/tools/testing/selftests/ipc/msgqueue.c
@@ -3,6 +3,7 @@
#include <string.h>
#include <errno.h>
#include <linux/msg.h>
+#include <fcntl.h>

#define MAX_MSG_SIZE 32

@@ -19,9 +20,9 @@ struct msg1 {
#define ANOTHER_MSG_TYPE 26538

struct msgqueue_data {
+ key_t key;
    int msq_id;
    int qbytes;
- int kern_id;
    int qnum;
    int mode;
    struct msg1 *messages;
@@ -29,41 +30,49 @@ struct msgqueue_data {

```

```

int restore_queue(struct msgque_data *msgque)
{
- struct msqid_ds ds;
- int id, i;
+ int fd, ret, id, i;
+ char buf[32];

- id = msgget(msgque->msq_id,
-     msgque->mode | IPC_CREAT | IPC_EXCL | IPC_PRESET);
- if (id == -1) {
-     printf("Failed to create queue\n");
+ fd = open("/proc/sys/kernel/msg_next_id", O_WRONLY);
+ if (fd == -1) {
+     printf("Failed to open /proc/sys/kernel/msg_next_id\n");
+     return -errno;
+ }
+ sprintf(buf, "%d", msgque->msq_id);

- if (id != msgque->msq_id) {
-     printf("Failed to preset id (%d instead of %d)\n",
-         id, msgque->msq_id);
-     return -EFAULT;
+ ret = write(fd, buf, strlen(buf));
+ if (ret != strlen(buf)) {
+     printf("Failed to write to /proc/sys/kernel/msg_next_id\n");
+     return -errno;
+ }

- if (msgctl(id, MSG_STAT, &ds) < 0) {
-     printf("Failed to stat queue\n");
+ id = msgget(msgque->key, msgque->mode | IPC_CREAT | IPC_EXCL);
+ if (id == -1) {
+     printf("Failed to create queue\n");
+     return -errno;
+ }

- ds.msg_perm.key = msgque->msq_id;
- ds.msg_qbytes = msgque->qbytes;
- if (msgctl(id, MSG_SET, &ds) < 0) {
-     printf("Failed to update message key\n");
-     return -errno;
+ if (id != msgque->msq_id) {
+     printf("Restored queue has wrong id (%d instead of %d)\n",
+         id, msgque->msq_id);
+     ret = -EFAULT;
+     goto destroy;
+ }

```

```

    for (i = 0; i < msgqueue->qnum; i++) {
-   if (msgsnd(msgqueue->msq_id, &msgqueue->messages[i].mtype, msgqueue->messages[i].msize,
IPC_NOWAIT) != 0) {
+   if (msgsnd(msgqueue->msq_id, &msgqueue->messages[i].mtype,
+   msgqueue->messages[i].msize, IPC_NOWAIT) != 0) {
        printf("msgsnd failed (%m)\n");
-   return -errno;
+   ret = -errno;
+   goto destroy;
    };
}
return 0;
+
+destroy:
+ if (msgctl(id, IPC_RMID, 0))
+ printf("Failed to destroy queue: %d\n", -errno);
+ return ret;
}

```

```

int check_and_destroy_queue(struct msgqueue_data *msgqueue)

```

```

@@ -72,7 +81,8 @@ int check_and_destroy_queue(struct msgqueue_data *msgqueue)
    int cnt = 0, ret;

```

```

    while (1) {
-   ret = msgrcv(msgqueue->msq_id, &message.mtype, MAX_MSG_SIZE, 0, IPC_NOWAIT);
+   ret = msgrcv(msgqueue->msq_id, &message.mtype, MAX_MSG_SIZE,
+   0, IPC_NOWAIT);
        if (ret < 0) {
            if (errno == ENOMSG)
                break;

```

```

@@ -81,7 +91,8 @@ int check_and_destroy_queue(struct msgqueue_data *msgqueue)
        goto err;
    }
    if (ret != msgqueue->messages[cnt].msize) {
-   printf("Wrong message size: %d (expected %d)\n", ret, msgqueue->messages[cnt].msize);
+   printf("Wrong message size: %d (expected %d)\n", ret,
+   msgqueue->messages[cnt].msize);
        ret = -EINVAL;
        goto err;
    }

```

```

@@ -115,15 +126,17 @@ err:

```

```

int dump_queue(struct msgqueue_data *msgqueue)
{
- struct msqid_ds ds;
+ struct msqid64_ds ds;
+ int kern_id;
    int i, ret;

```

```

- for (msgqueue->kern_id = 0; msgqueue->kern_id < 256; msgqueue->kern_id++) {
- ret = msgctl(msgqueue->kern_id, MSG_STAT, &ds);
+ for (kern_id = 0; kern_id < 256; kern_id++) {
+ ret = msgctl(kern_id, MSG_STAT, &ds);
  if (ret < 0) {
    if (errno == -EINVAL)
      continue;
- printf("Failed to get stats for IPC queue with id %d\n", msgqueue->kern_id);
+ printf("Failed to get stats for IPC queue with id %d\n",
+ kern_id);
  return -errno;
}

```

```

@@ -142,7 +155,8 @@ int dump_queue(struct msgqueue_data *msgqueue)
  msgqueue->qbytes = ds.msg_qbytes;

```

```

  for (i = 0; i < msgqueue->qnum; i++) {
- ret = msgrcv(msgqueue->msq_id, &msgqueue->messages[i].mtype, MAX_MSG_SIZE, i,
IPC_NOWAIT | MSG_COPY);
+ ret = msgrcv(msgqueue->msq_id, &msgqueue->messages[i].mtype,
+ MAX_MSG_SIZE, i, IPC_NOWAIT | MSG_COPY);
  if (ret < 0) {
    printf("Failed to copy IPC message: %m (%d)\n", errno);
    return -errno;
@@ -158,33 +172,34 @@ int fill_msgqueue(struct msgqueue_data *msgqueue)

```

```

  msgbuf.mtype = MSG_TYPE;
  memcpy(msgbuf.mtext, TEST_STRING, sizeof(TEST_STRING));
- if (msgsnd(msgqueue->msq_id, &msgbuf.mtype, sizeof(TEST_STRING), IPC_NOWAIT) != 0) {
+ if (msgsnd(msgqueue->msq_id, &msgbuf.mtype, sizeof(TEST_STRING),
+ IPC_NOWAIT) != 0) {
  printf("First message send failed (%m)\n");
  return -errno;
};

```

```

  msgbuf.mtype = ANOTHER_MSG_TYPE;
  memcpy(msgbuf.mtext, ANOTHER_TEST_STRING, sizeof(ANOTHER_TEST_STRING));
- if (msgsnd(msgqueue->msq_id, &msgbuf.mtype, sizeof(ANOTHER_TEST_STRING),
IPC_NOWAIT) != 0) {
+ if (msgsnd(msgqueue->msq_id, &msgbuf.mtype, sizeof(ANOTHER_TEST_STRING),
+ IPC_NOWAIT) != 0) {
  printf("Second message send failed (%m)\n");
  return -errno;
};
return 0;
}

```

```
-int main (int argc, char **argv)
+int main(int argc, char **argv)
{
- key_t key;
  int msg, pid, err;
  struct msgque_data msgque;

- key = ftok(argv[0], 822155650);
- if (key == -1) {
+ msgque.key = ftok(argv[0], 822155650);
+ if (msgque.key == -1) {
    printf("Can't make key\n");
    return -errno;
  }

- msgque.msq_id = msgget(key, IPC_CREAT | IPC_EXCL | 0666);
+ msgque.msq_id = msgget(msgque.key, IPC_CREAT | IPC_EXCL | 0666);
  if (msgque.msq_id == -1) {
    printf("Can't create queue\n");
    goto err_out;
  }
}
```
