Subject: [PATCH 3/4] - v2 - IPC: use the target ID specified in procfs
Posted by Nadia Derbey on Fri, 18 Apr 2008 05:45:02 GMT
View Forum Message <> Reply to Message

[PATCH 03/04]

This patch makes use of the target id specified by a previous write into
/proc/self/task/<tid>/next_id as the id to use to allocate the next IPC
object.

Signed-off-by: Nadia Derbey <Nadia.Derbey@bull.net>

---
 include/linux/sysids.h | 7 +++++++
 ipc/util.c | 41 ++++++++++++++++++++++++++++++++++++++---------
 kernel/nextid.c | 2 +-
 3 files changed, 41 insertions(+), 9 deletions(-)

Index: linux-2.6.25-rc8-mm2/include/linux/sysids.h
===
--- linux-2.6.25-rc8-mm2.orig/include/linux/sysids.h 2008-04-17 16:03:07.000000000 +0200
+++ linux-2.6.25-rc8-mm2/include/linux/sysids.h 2008-04-17 17:05:54.000000000 +0200
@@ -37,9 +37,16 @@ struct sys_id {
  long *blocks[0];
 };

+#define next_ipcid(tsk) ((tsk)->next_id   \
+   ? ((tsk)->next_id->nids   \
+   ? ID_AT((tsk)->next_id, 0) \
+   : -1)   \
+   : -1)
+
 extern ssize_t get_nextid(struct task_struct *, char *, size_t);
 extern int set_nextid(struct task_struct *, char *);
 extern int reset_nextid(struct task_struct *);
+extern void id_blocks_free(struct sys_id *);

 static inline void exit_nextid(struct task_struct *tsk)
 {
Index: linux-2.6.25-rc8-mm2/kernel/nextid.c
===
--- linux-2.6.25-rc8-mm2.orig/kernel/nextid.c 2008-04-17 16:54:30.000000000 +0200
+++ linux-2.6.25-rc8-mm2/kernel/nextid.c 2008-04-17 17:06:43.000000000 +0200
@@ -49,7 +49,7 @@ out_undo_partial_alloc:
  return NULL;
 }

-static void id_blocks_free(struct sys_id *ids)

```
+void id_blocks_free(struct sys_id *ids)
 {
  if (ids == NULL)
   return;
Index: linux-2.6.25-rc8-mm2/ipc/util.c
===================================================================
--- linux-2.6.25-rc8-mm2.orig/ipc/util.c 2008-04-17 12:50:37.000000000 +0200
+++ linux-2.6.25-rc8-mm2/ipc/util.c 2008-04-17 17:09:37.000000000 +0200
@@ -260,6 +260,7 @@ int ipc_get_maxid(struct ipc_ids *ids)
 int ipc_addid(struct ipc_ids* ids, struct kern_ipc_perm* new, int size)
 {
  int id, err;
+ int next_id;

  if (size > IPCMNI)
   size = IPCMNI;
@@ -267,20 +268,44 @@ int ipc_addid(struct ipc_ids* ids, struc
  if (ids->in_use >= size)
   return -ENOSPC;

- err = idr_get_new(&ids->ipcs_idr, new, &id);
- if (err)
-  return err;
+ next_id = next_ipcid(current);
+
+ if (next_id >= 0) {
+  /* There is a target id specified, try to use it */
+  int new_lid = next_id % SEQ_MULTIPLIER;
+
+  if (next_id !=
+      (new_lid + (next_id / SEQ_MULTIPLIER) * SEQ_MULTIPLIER))
+   return -EINVAL;
+
+  err = idr_get_new_above(&ids->ipcs_idr, new, new_lid, &id);
+  if (err)
+   return err;
+  if (id != new_lid) {
+   idr_remove(&ids->ipcs_idr, id);
+   return -EBUSY;
+  }
+
+  new->id = next_id;
+  new->seq = next_id / SEQ_MULTIPLIER;
+  id_blocks_free(current->next_id);
+  current->next_id = NULL;
+ } else {
+  err = idr_get_new(&ids->ipcs_idr, new, &id);
+  if (err)
```

```
+   return err;
+
+  new->seq = ids->seq++;
+  if (ids->seq > ids->seq_max)
+   ids->seq = 0;
+  new->id = ipc_buildid(id, new->seq);
+ }

  ids->in_use++;

  new->cuid = new->uid = current->euid;
  new->gid = new->cgid = current->egid;

- new->seq = ids->seq++;
- if(ids->seq > ids->seq_max)
-  ids->seq = 0;
-
- new->id = ipc_buildid(id, new->seq);
  spin_lock_init(&new->lock);
  new->deleted = 0;
  rcu_read_lock();

--
```