Subject: [PATCH I2O] memory leak in i2o_exec_lct_modified
Posted by vaverin on Sat, 04 Mar 2006 08:34:45 GMT
View Forum Message <> Reply to Message

i2o_exec_lct_modified() does not release memory allocated for work_struct.

Signed-off-by: Vasily Averin <vvs@sw.ru>

Thank you,
Vasily Averin

SWsoft Virtuozzo/OpenVZ Linux kernel team

```
--- ./drivers/message/i2o/exec-osm.c.i2ml 2006-03-04 11:09:45.000000000 +0300
+++ ./drivers/message/i2o/exec-osm.c 2006-03-04 11:09:03.000000000 +0300
@@ -57,6 +57,11 @@ struct i2o_exec_wait {
  struct list_head list; /* node in global wait list */
 };

+struct i2o_workqueue {
+ struct work_struct work;
+ struct i2o_controller *c;
+};
+
 /* Exec OSM class handling definition */
 static struct i2o_class_id i2o_exec_class_id[] = {
  {I2O_CLASS_EXECUTIVE},
@@ -355,16 +360,19 @@ static int i2o_exec_remove(struct device
  * new LCT and if the buffer for the LCT was to small sends a LCT NOTIFY
  * again, otherwise send LCT NOTIFY to get informed on next LCT change.
  */
-static void i2o_exec_lct_modified(struct i2o_controller *c)
+static void i2o_exec_lct_modified(void *data)
 {
  u32 change_ind = 0;
+ struct i2o_workqueue *cp;

- if (i2o_device_parse_lct(c) != -EAGAIN)
-  change_ind = c->lct->change_ind + 1;
+ cp = (struct i2o_workqueue *)data;
+ if (i2o_device_parse_lct(cp->c) != -EAGAIN)
+  change_ind = cp->c->lct->change_ind + 1;

 #ifdef CONFIG_I2O_LCT_NOTIFY_ON_CHANGES
- i2o_exec_lct_notify(c, change_ind);
+ i2o_exec_lct_notify(cp->c, change_ind);
 #endif
```

```
+ kfree(cp);
 };

 /**
@@ -410,16 +418,22 @@ static int i2o_exec_reply(struct i2o_con
   return i2o_msg_post_wait_complete(c, m, msg, context);

  if ((le32_to_cpu(msg->u.head[1]) >> 24) == I2O_CMD_LCT_NOTIFY) {
-  struct work_struct *work;
+  struct i2o_workqueue *cp;

   pr_debug("%s: LCT notify received\n", c->name);

-  work = kmalloc(sizeof(*work), GFP_ATOMIC);
-  if (!work)
+  cp = kmalloc(sizeof(struct i2o_workqueue), GFP_ATOMIC);
+  if (!cp)
    return -ENOMEM;

-  INIT_WORK(work, (void (*)(void *))i2o_exec_lct_modified, c);
-  queue_work(i2o_exec_driver.event_queue, work);
+  cp->c = c;
+  INIT_WORK(&cp->work, i2o_exec_lct_modified, cp);
+  if (!queue_work(i2o_exec_driver.event_queue, &cp->work)) {
+   printk(KERN_DEBUG "i2o_exec_reply:"
+    " call to queue_work() failed.\n");
+   kfree(cp);
+   return -EIO;
+  }
   return 1;
  }
```