Subject: Re: [PATCH] bluetooth rfcomm: Convert to kthread API.
Posted by Christoph Hellwig on Sun, 22 Apr 2007 20:14:27 GMT
View Forum Message <> Reply to Message

On Thu, Apr 19, 2007 at 04:12:53PM -0700, Andrew Morton wrote:
> On Thu, 19 Apr 2007 01:58:54 -0600
> "Eric W. Biederman" <ebiederm@xmission.com> wrote:
>
> > From: Eric W. Biederman <ebiederm@xmission.com>
> >
> > This patch starts krfcommd using kthread_run instead of a combination
> > of kernel_thread and daemonize making the code slightly simpler
> > and more maintainable.
>
> gargh, the more I look at these things, the more I agree with Christoph.

Hehe. Here's a patch to do the full kthread conversion for rfcomm, it
doesn't have the asynchrnous termination issues the other bluetooth drivers
have. Also handle init failures in rfcomm while we're at it.

Signed-off-by: Christoph Hellwig <hch@lst.de>

Index: linux-2.6/net/bluetooth/rfcomm/core.c
==========
--- linux-2.6.orig/net/bluetooth/rfcomm/core.c 2007-04-22 21:01:31.000000000 +0200
+++ linux-2.6/net/bluetooth/rfcomm/core.c 2007-04-22 21:12:30.000000000 +0200
@@ -37,6 +37,7 @@
 #include <linux/device.h>
 #include <linux/net.h>
 #include <linux/mutex.h>
+#include <linux/kthread.h>

 #include <net/sock.h>
 #include <asm/uaccess.h>
@@ -67,7 +68,6 @@ static DEFINE_MUTEX(rfcomm_mutex);
 static unsigned long rfcomm_event;

 static LIST_HEAD(session_list);
-static atomic_t terminate, running;

 static int rfcomm_send_frame(struct rfcomm_session *s, u8 *data, int len);
 static int rfcomm_send_sabm(struct rfcomm_session *s, u8 dlci);
@@ -1846,26 +1846,6 @@ static inline void rfcomm_process_sessio
  rfcomm_unlock();
 }

-static void rfcomm_worker(void)

```
-{
- BT_DBG("");
-
- while (!atomic_read(&terminate)) {
-  if (!test_bit(RFCOMM_SCHED_WAKEUP, &rfcomm_event)) {
-   /* No pending events. Let's sleep.
-    * Incoming connections and data will wake us up. */
-   set_current_state(TASK_INTERRUPTIBLE);
-   schedule();
-  }
-
-  /* Process stuff */
-  clear_bit(RFCOMM_SCHED_WAKEUP, &rfcomm_event);
-  rfcomm_process_sessions();
- }
- set_current_state(TASK_RUNNING);
- return;
-}
-
 static int rfcomm_add_listener(bdaddr_t *ba)
 {
  struct sockaddr_l2 addr;
@@ -1931,23 +1911,27 @@ static void rfcomm_kill_listener(void)

 static int rfcomm_run(void *unused)
 {
- rfcomm_thread = current;
-
- atomic_inc(&running);
-
- daemonize("krfcommd");
  set_user_nice(current, -10);
  current->flags |= PF_NOFREEZE;

  BT_DBG("");

  rfcomm_add_listener(BDADDR_ANY);
+ while (!kthread_should_stop()) {
+  if (!test_bit(RFCOMM_SCHED_WAKEUP, &rfcomm_event)) {
+   /* No pending events. Let's sleep.
+    * Incoming connections and data will wake us up. */
+   set_current_state(TASK_INTERRUPTIBLE);
+   schedule();
+  }

- rfcomm_worker();
-
+  /* Process stuff */
```

```
+  clear_bit(RFCOMM_SCHED_WAKEUP, &rfcomm_event);
+  rfcomm_process_sessions();
+ }
+ set_current_state(TASK_RUNNING);
  rfcomm_kill_listener();

- atomic_dec(&running);
  return 0;
 }

@@ -2052,24 +2036,52 @@ static CLASS_ATTR(rfcomm_dlc, S_IRUGO, r
 /* ---- Initialization ---- */
 static int __init rfcomm_init(void)
 {
+ int err;
+
  l2cap_load();

- hci_register_cb(&rfcomm_cb);
+ err = hci_register_cb(&rfcomm_cb);
+ if (err)
+  goto out;

- kernel_thread(rfcomm_run, NULL, CLONE_KERNEL);
+ rfcomm_thread = kthread_run(rfcomm_run, NULL, "krfcommd");
+ if (IS_ERR(rfcomm_thread)) {
+  err = PTR_ERR(rfcomm_thread);
+  goto out_unregister_hci;
+ }

- if (class_create_file(bt_class, &class_attr_rfcomm_dlc) < 0)
+ err = class_create_file(bt_class, &class_attr_rfcomm_dlc);
+ if (err < 0) {
   BT_ERR("Failed to create RFCOMM info file");
+  goto out_kthread_stop;
+ }

- rfcomm_init_sockets();
+ err = rfcomm_init_sockets();
+ if (err)
+  goto out_remove_sysfs_files;

 #ifdef CONFIG_BT_RFCOMM_TTY
- rfcomm_init_ttys();
+ err = rfcomm_init_ttys();
+ if (err)
+  goto out_cleanup_sockets;
 #endif
```

```
  BT_INFO("RFCOMM ver %s", VERSION);

  return 0;
+
+#ifdef CONFIG_BT_RFCOMM_TTY
+ out_cleanup_sockets:
+ rfcomm_cleanup_sockets();
+#endif
+ out_remove_sysfs_files:
+ class_remove_file(bt_class, &class_attr_rfcomm_dlc);
+ out_unregister_hci:
+ hci_unregister_cb(&rfcomm_cb);
+ out_kthread_stop:
+ kthread_stop(rfcomm_thread);
+ out:
+ return err;
 }

 static void __exit rfcomm_exit(void)
@@ -2077,15 +2089,7 @@ static void __exit rfcomm_exit(void)
  class_remove_file(bt_class, &class_attr_rfcomm_dlc);

  hci_unregister_cb(&rfcomm_cb);
-
- /* Terminate working thread.
-  * ie. Set terminate flag and wake it up */
- atomic_inc(&terminate);
- rfcomm_schedule(RFCOMM_SCHED_STATE);
-
- /* Wait until thread is running */
- while (atomic_read(&running))
-  schedule();
+ kthread_stop(rfcomm_thread);

 #ifdef CONFIG_BT_RFCOMM_TTY
  rfcomm_cleanup_ttys();
```

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers