Subject: [PATCH] net/rxrpc: Convert to kthread API.
Posted by ebiederm on Thu, 19 Apr 2007 07:58:56 GMT
View Forum Message <> Reply to Message

From: Eric W. Biederman <ebiederm@xmission.com>

This patch modifies the startup of krxtimod, krxiod, and krxsecd
to use kthread_run instead of a combination of kernel_thread
and daemonize making the code slightly simpler and more maintainable.

In addition since by default all signals are ignored when delivered
to a kernel thread the code to flush signals has been removed.

Cc: David Howells <dhowells@redhat.com>
Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>
---
 net/rxrpc/internal.h | 11 -----------
 net/rxrpc/krxiod.c | 16 ++++++++--------
 net/rxrpc/krxsecd.c | 16 ++++++++--------
 net/rxrpc/krxtimod.c | 15 ++++++++-------
 4 files changed, 22 insertions(+), 36 deletions(-)

diff --git a/net/rxrpc/internal.h b/net/rxrpc/internal.h
index cc0c579..1dd69aa 100644
--- a/net/rxrpc/internal.h
+++ b/net/rxrpc/internal.h
@@ -49,17 +49,6 @@ __RXACCT_DECL(extern atomic_t rxrpc_message_count);
 #define _net(FMT, a...) do { if (rxrpc_knet)   knet (FMT , ##a); } while(0)
 #endif

-static inline void rxrpc_discard_my_signals(void)
-{
- while (signal_pending(current)) {
- siginfo_t sinfo;
-
- spin_lock_irq(&current->sighand->siglock);
- dequeue_signal(current, &current->blocked, &sinfo);
- spin_unlock_irq(&current->sighand->siglock);
- }
-}
-
 /*
  * call.c
  */
diff --git a/net/rxrpc/krxiod.c b/net/rxrpc/krxiod.c
index bbbcd6c..c590ccd 100644
--- a/net/rxrpc/krxiod.c
+++ b/net/rxrpc/krxiod.c

```
@@ -14,6 +14,7 @@
 #include <linux/spinlock.h>
 #include <linux/init.h>
 #include <linux/freezer.h>
+#include <linux/kthread.h>
 #include <rxrpc/krxiod.h>
 #include <rxrpc/transport.h>
 #include <rxrpc/peer.h>
@@ -43,8 +44,6 @@ static int rxrpc_krxiod(void *arg)

  printk("Started krxiod %d\n",current->pid);

- daemonize("krxiod");
-
  /* loop around waiting for work to do */
  do {
   /* wait for work or to be told to exit */
@@ -57,8 +56,7 @@ static int rxrpc_krxiod(void *arg)
    for (;;) {
     set_current_state(TASK_INTERRUPTIBLE);
     if (atomic_read(&rxrpc_krxiod_qcount) ||
-        rxrpc_krxiod_die ||
-        signal_pending(current))
+        rxrpc_krxiod_die)
      break;

     schedule();
@@ -141,9 +139,6 @@ static int rxrpc_krxiod(void *arg)

   try_to_freeze();

-  /* discard pending signals */
-  rxrpc_discard_my_signals();
-
  } while (!rxrpc_krxiod_die);

  /* and that's all */
@@ -157,7 +152,12 @@ static int rxrpc_krxiod(void *arg)
  */
 int __init rxrpc_krxiod_init(void)
 {
- return kernel_thread(rxrpc_krxiod, NULL, 0);
+ struct task_struct *task;
+ int ret = 0;
+ task = kthread_run(rxrpc_krxiod, NULL, "krxiod");
+ if (IS_ERR(task))
+  ret = PTR_ERR(task);
+ return ret;
```

```
 } /* end rxrpc_krxiod_init() */

diff --git a/net/rxrpc/krxsecd.c b/net/rxrpc/krxsecd.c
index 9a1e7f5..150cd39 100644
--- a/net/rxrpc/krxsecd.c
+++ b/net/rxrpc/krxsecd.c
@@ -19,6 +19,7 @@
 #include <linux/completion.h>
 #include <linux/spinlock.h>
 #include <linux/init.h>
+#include <linux/kthread.h>
 #include <rxrpc/krxsecd.h>
 #include <rxrpc/transport.h>
 #include <rxrpc/connection.h>
@@ -56,8 +57,6 @@ static int rxrpc_krxsecd(void *arg)

  printk("Started krxsecd %d\n", current->pid);

- daemonize("krxsecd");
-
  /* loop around waiting for work to do */
  do {
   /* wait for work or to be told to exit */
@@ -70,8 +69,7 @@ static int rxrpc_krxsecd(void *arg)
    for (;;) {
     set_current_state(TASK_INTERRUPTIBLE);
     if (atomic_read(&rxrpc_krxsecd_qcount) ||
-        rxrpc_krxsecd_die ||
-        signal_pending(current))
+        rxrpc_krxsecd_die)
      break;

     schedule();
@@ -110,9 +108,6 @@ static int rxrpc_krxsecd(void *arg)

   try_to_freeze();

-  /* discard pending signals */
-  rxrpc_discard_my_signals();
-
  } while (!die);

  /* and that's all */
@@ -126,7 +121,12 @@ static int rxrpc_krxsecd(void *arg)
  */
 int __init rxrpc_krxsecd_init(void)
 {
```

```
- return kernel_thread(rxrpc_krxsecd, NULL, 0);
+ struct task_struct *task;
+ int ret = 0;
+ task = kthread_run(rxrpc_krxsecd, NULL, "krxsecd");
+ if (IS_ERR(task))
+  ret = PTR_ERR(task);
+ return ret;

 } /* end rxrpc_krxsecd_init() */

diff --git a/net/rxrpc/krxtimod.c b/net/rxrpc/krxtimod.c
index 9a9b613..3b5f062 100644
--- a/net/rxrpc/krxtimod.c
+++ b/net/rxrpc/krxtimod.c
@@ -14,6 +14,7 @@
 #include <linux/sched.h>
 #include <linux/completion.h>
 #include <linux/freezer.h>
+#include <linux/kthread.h>
 #include <rxrpc/rxrpc.h>
 #include <rxrpc/krxtimod.h>
 #include <asm/errno.h>
@@ -35,11 +36,12 @@ static int krxtimod(void *arg);
  */
 int rxrpc_krxtimod_start(void)
 {
- int ret;
+ struct task_struct *task;
+ int ret = 0;

- ret = kernel_thread(krxtimod, NULL, 0);
- if (ret < 0)
-  return ret;
+ task = kthread_run(krxtimod, NULL, "krxtimod");
+ if (IS_ERR(task))
+  ret = PTR_ERR(task);

  wait_for_completion(&krxtimod_alive);

@@ -71,8 +73,6 @@ static int krxtimod(void *arg)

  printk("Started krxtimod %d\n", current->pid);

- daemonize("krxtimod");
-
  complete(&krxtimod_alive);

  /* loop around looking for things to attend to */
```

```
@@ -93,9 +93,6 @@ static int krxtimod(void *arg)

   try_to_freeze();

-  /* discard pending signals */
-  rxrpc_discard_my_signals();
-
   /* work out the time to elapse before the next event */
   spin_lock(&krxtimod_lock);
   if (list_empty(&krxtimod_list)) {
--
1.5.0.g53756
```