Subject: Re: [PATCH] mtd_blkdevs: Convert to use the kthread API
Posted by Christoph Hellwig on Sun, 22 Apr 2007 19:40:57 GMT
View Forum Message <> Reply to Message

On Sun, Apr 22, 2007 at 01:24:53PM +0100, Christoph Hellwig wrote:
> On Thu, Apr 19, 2007 at 12:55:28AM -0600, Eric W. Biederman wrote:
> > From: Eric W. Biederman <ebiederm@xmission.com> - unquoted
> >
> > thread_run is used intead of kernel_thread, daemonize, and mucking
> > around blocking signals directly.
>
> This is the full conversion I sent to Dave in April 2006, but never got
> any feedback to:

Here's a slightly updated version that corrects the set_current_state
placement as discussed with Dave on irc:

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

Index: linux-2.6/drivers/mtd/mtd_blkdevs.c
==========
--- linux-2.6.orig/drivers/mtd/mtd_blkdevs.c 2007-01-29 10:03:52.000000000 +0100
+++ linux-2.6/drivers/mtd/mtd_blkdevs.c 2007-04-22 20:39:20.000000000 +0200
@@ -20,6 +20,7 @@
 #include <linux/hdreg.h>
 #include <linux/init.h>
 #include <linux/mutex.h>
+#include <linux/kthread.h>
 #include <asm/uaccess.h>

 static LIST_HEAD(blktrans_majors);
@@ -28,9 +29,7 @@ extern struct mutex mtd_table_mutex;
 extern struct mtd_info *mtd_table[];

 struct mtd_blkcore_priv {
- struct completion thread_dead;
- int exiting;
- wait_queue_head_t thread_wq;
+ struct task_struct *thread;
  struct request_queue *rq;
  spinlock_t queue_lock;
 };
@@ -83,38 +82,19 @@ static int mtd_blktrans_thread(void *arg
  /* we might get involved when memory gets low, so use PF_MEMALLOC */
  current->flags |= PF_MEMALLOC | PF_NOFREEZE;

- daemonize("%sd", tr->name);

```
-
- /* daemonize() doesn't do this for us since some kernel threads
-    actually want to deal with signals. We can't just call
-    exit_sighand() since that'll cause an oops when we finally
-    do exit. */
- spin_lock_irq(&current->sighand->siglock);
- sigfillset(&current->blocked);
- recalc_sigpending();
- spin_unlock_irq(&current->sighand->siglock);
-
  spin_lock_irq(rq->queue_lock);
-
- while (!tr->blkcore_priv->exiting) {
+ while (!kthread_should_stop()) {
   struct request *req;
   struct mtd_blktrans_dev *dev;
   int res = 0;
-  DECLARE_WAITQUEUE(wait, current);

   req = elv_next_request(rq);

   if (!req) {
-   add_wait_queue(&tr->blkcore_priv->thread_wq, &wait);
    set_current_state(TASK_INTERRUPTIBLE);
-
    spin_unlock_irq(rq->queue_lock);
-
    schedule();
-   remove_wait_queue(&tr->blkcore_priv->thread_wq, &wait);
-
    spin_lock_irq(rq->queue_lock);
-
    continue;
   }

@@ -133,13 +113,13 @@ static int mtd_blktrans_thread(void *arg
  }
  spin_unlock_irq(rq->queue_lock);

- complete_and_exit(&tr->blkcore_priv->thread_dead, 0);
+ return 0;
 }

 static void mtd_blktrans_request(struct request_queue *rq)
 {
  struct mtd_blktrans_ops *tr = rq->queuedata;
- wake_up(&tr->blkcore_priv->thread_wq);
+ wake_up_process(tr->blkcore_priv->thread);
```

```
 }


@@ -388,8 +368,6 @@ int register_mtd_blktrans(struct mtd_blk
   return ret;
  }
  spin_lock_init(&tr->blkcore_priv->queue_lock);
- init_completion(&tr->blkcore_priv->thread_dead);
- init_waitqueue_head(&tr->blkcore_priv->thread_wq);

  tr->blkcore_priv->rq = blk_init_queue(mtd_blktrans_request, &tr->blkcore_priv->queue_lock);
  if (!tr->blkcore_priv->rq) {
@@ -403,13 +381,14 @@ int register_mtd_blktrans(struct mtd_blk
  blk_queue_hardsect_size(tr->blkcore_priv->rq, tr->blksize);
  tr->blkshift = ffs(tr->blksize) - 1;

- ret = kernel_thread(mtd_blktrans_thread, tr, CLONE_KERNEL);
- if (ret < 0) {
+ tr->blkcore_priv->thread = kthread_run(mtd_blktrans_thread, tr,
+   "%sd", tr->name);
+ if (IS_ERR(tr->blkcore_priv->thread)) {
   blk_cleanup_queue(tr->blkcore_priv->rq);
   unregister_blkdev(tr->major, tr->name);
   kfree(tr->blkcore_priv);
   mutex_unlock(&mtd_table_mutex);
-  return ret;
+  return PTR_ERR(tr->blkcore_priv->thread);
  }

  INIT_LIST_HEAD(&tr->devs);
@@ -432,9 +411,7 @@ int deregister_mtd_blktrans(struct mtd_b
  mutex_lock(&mtd_table_mutex);

  /* Clean up the kernel thread */
- tr->blkcore_priv->exiting = 1;
- wake_up(&tr->blkcore_priv->thread_wq);
- wait_for_completion(&tr->blkcore_priv->thread_dead);
+ kthread_stop(tr->blkcore_priv->thread);

  /* Remove it from the list of active majors */
  list_del(&tr->list);
```

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