
Subject: [PATCH] fs/afs: Convert to kthread API.
Posted by [ebiederm](#) on Thu, 19 Apr 2007 07:58:55 GMT
[View Forum Message](#) <> [Reply to Message](#)

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

This patch modifies the startup of kafscmd, kafsasyncd, and kafstimod 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>

```
fs/afs/cmsservice.c | 10 +++++-----
fs/afs/internal.h   | 11 -----
fs/afs/kafsasyncd.c | 17 ++++++-----
fs/afs/kafstimod.c  | 16 ++++++-----
4 files changed, 17 insertions(+), 37 deletions(-)
```

```
diff --git a/fs/afs/cmsservice.c b/fs/afs/cmsservice.c
```

```
index 3d097fd..f7e2355 100644
```

```
--- a/fs/afs/cmsservice.c
```

```
+++ b/fs/afs/cmsservice.c
```

```
@@ -13,6 +13,7 @@
```

```
#include <linux/init.h>
```

```
#include <linux/sched.h>
```

```
#include <linux/completion.h>
```

```
+#include <linux/kthread.h>
```

```
#include "server.h"
```

```
#include "cell.h"
```

```
#include "transport.h"
```

```
@@ -120,8 +121,6 @@ static int kafscmd(void *arg)
```

```
    printk(KERN_INFO "kAFS: Started kafscmd %d\n", current->pid);
```

```
-    daemonize("kafscmd");
```

```
-
```

```
    complete(&kafscmd_alive);
```

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

```
@@ -133,7 +132,6 @@ static int kafscmd(void *arg)
```

```
    for (;;) {
```

```
        set_current_state(TASK_INTERRUPTIBLE);
```

```
        if (!list_empty(&kafscmd_attention_list) ||
```

```
-        signal_pending(current) ||
```

```
kafscmd_die)
break;
```

```
@@ -297,8 +295,10 @@ int afscm_start(void)
```

```
    down_write(&afscm_sem);
    if (!afscm_usage) {
-   ret = kernel_thread(kafscmd, NULL, 0);
-   if (ret < 0)
+   struct task_struct *task;
+   task = kthread_run(kafscmd, NULL, "kafscmd");
+   ret = PTR_ERR(task);
+   if (IS_ERR(task))
        goto out;
```

```
    wait_for_completion(&kafscmd_alive);
diff --git a/fs/afs/internal.h b/fs/afs/internal.h
index 5151d5d..2d667b7 100644
--- a/fs/afs/internal.h
+++ b/fs/afs/internal.h
@@ -40,17 +40,6 @@
#define _net(FMT, a...) do { } while(0)
#endif
```

```
-static inline void afs_discard_my_signals(void)
-{
-   while (signal_pending(current)) {
-       siginfo_t sinfo;
-
-       spin_lock_irq(&current->sigband->siglock);
-       dequeue_signal(current, &current->blocked, &sinfo);
-       spin_unlock_irq(&current->sigband->siglock);
-   }
-}
-
/*
 * cell.c
 */
```

```
diff --git a/fs/afs/kafsasyncd.c b/fs/afs/kafsasyncd.c
index 615df24..ead025f 100644
--- a/fs/afs/kafsasyncd.c
+++ b/fs/afs/kafsasyncd.c
@@ -21,6 +21,7 @@
#include <linux/sched.h>
#include <linux/completion.h>
#include <linux/freezer.h>
+#include <linux/kthread.h>
#include "cell.h"
```

```

#include "server.h"
#include "volume.h"
@@ -56,15 +57,15 @@ static void kafsasyncd_null_call_error_func(struct rxrpc_call *call)
    */
int afs_kafsasyncd_start(void)
{
- int ret;
+ struct task_struct *task;

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

    wait_for_completion(&kafsasyncd_alive);

- return ret;
+ return 0;
} /* end afs_kafsasyncd_start() */

/*****
@@ -95,8 +96,6 @@ static int kafsasyncd(void *arg)

    printk("kAFS: Started kafsasyncd %d\n", current->pid);

- daemonize("kafsasyncd");
-
    complete(&kafsasyncd_alive);

    /* loop around looking for things to attend to */
@@ -106,7 +105,6 @@ static int kafsasyncd(void *arg)

    for (;;) {
        if (!list_empty(&kafsasyncd_async_attnq) ||
-         signal_pending(current) ||
            kafsasyncd_die)
            break;

@@ -119,9 +117,6 @@ static int kafsasyncd(void *arg)

    try_to_freeze();

- /* discard pending signals */
- afs_discard_my_signals();
-
    die = kafsasyncd_die;

```

```

/* deal with the next asynchronous operation requiring
diff --git a/fs/afs/kafstimod.c b/fs/afs/kafstimod.c
index 694344e..caeac88 100644
--- a/fs/afs/kafstimod.c
+++ b/fs/afs/kafstimod.c
@@ -14,6 +14,7 @@
#include <linux/sched.h>
#include <linux/completion.h>
#include <linux/freezer.h>
+#include <linux/kthread.h>
#include "cell.h"
#include "volume.h"
#include "kafstimod.h"
@@ -36,15 +37,15 @@ static int kafstimod(void *arg);
*/
int afs_kafstimod_start(void)
{
- int ret;
+ struct task_struct *task;

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

wait_for_completion(&kafstimod_alive);

- return ret;
+ return 0;
} /* end afs_kafstimod_start() */

/*****
@@ -72,8 +73,6 @@ static int kafstimod(void *arg)

printf("kAFS: Started kafstimod %d\n", current->pid);

- daemonize("kafstimod");
-
complete(&kafstimod_alive);

/* loop around looking for things to attend to */
@@ -94,9 +93,6 @@ static int kafstimod(void *arg)

try_to_freeze();

```

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

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