

[PATCH 2/2] uml: rename and improve actually_do_remove()

Source: <http://linux.derkeiler.com/Mailing-Lists/Kernel/2006-04/msg06731.html>

- From: Paolo 'Blaisorblade' Giarrusso <blaisorblade@xxxxxxxx>
- Date: Sun, 30 Apr 2006 17:36:37 +0200

From: Paolo 'Blaisorblade' Giarrusso <blaisorblade@xxxxxxxx>

Rename actually_do_remove() to remove_files_and_dir(), make it call closedir(), make it ignore ENOENT (I see it frequently enough).

ENOENT is probably due to multiple threads calling the exitcall functions together*, but fixing that is non-trivial; and ignoring it is perfectly ok in any case.

* it can surely happen: last_ditch_exit() is installed as SIGTERM handler at boot, and it's not removed on thread creation. So killall vmlinux (which I do) surely causes that. I've seen also a crash which seems to do the same.

Installing the handler on only the main thread would make UML do no cleanup when another thread exits, and we're not sure we want that. And mutual exclusion in that context is tricky – we can't use spinlock in code not on a kernel stack (spinlock debugging uses "current" a lot).

Signed-off-by: Paolo 'Blaisorblade' Giarrusso <blaisorblade@xxxxxxxx>

arch/um/os-Linux/umid.c | 53 ++++++-----
1 files changed, 37 insertions(+), 16 deletions(-)

```
diff --git a/arch/um/os-Linux/umid.c b/arch/um/os-Linux/umid.c
index 7c53be1..8ef150f 100644
--- a/arch/um/os-Linux/umid.c
+++ b/arch/um/os-Linux/umid.c
@@ -67,32 +67,53 @@ err:
return err;
}
```

```
-static int actually_do_remove(char *dir)
+/*
+ * Unlinks the files contained in @dir and then removes @dir.
+ * Doesn't handle directory trees, so it's not like rm -rf, but almost such. We
+ * ignore ENOENT errors for anything (they happen, strangely enough – possibly due
+ * to races between multiple dying UML threads).
```

[PATCH 2/2] uml: rename and improve actually_do_remove()

```
+ */
+static int remove_files_and_dir(char *dir)
{
DIR *directory;
struct dirent *ent;
int len;
char file[256];
+ int ret;

directory = opendir(dir);
- if(directory == NULL)
- return -errno;
+ if (directory == NULL) {
+ if (errno != ENOENT)
+ return -errno;
+ else
+ return 0;
+ }

- while((ent = readdir(directory)) != NULL){
- if(!strcmp(ent->d_name, ".") || !strcmp(ent->d_name, ".."))
+ while ((ent = readdir(directory)) != NULL) {
+ if (!strcmp(ent->d_name, ".") || !strcmp(ent->d_name, ".."))
continue;
len = strlen(dir) + sizeof("/") + strlen(ent->d_name) + 1;
- if(len > sizeof(file))
- return -E2BIG;
+ if (len > sizeof(file)) {
+ ret = -E2BIG;
+ goto out;
+ }

sprintf(file, "%s/%s", dir, ent->d_name);
- if(unlink(file) < 0)
- return -errno;
+ if (unlink(file) < 0 && errno != ENOENT) {
+ ret = -errno;
+ goto out;
+ }
}
- if(rmdir(dir) < 0)
- return -errno;

- return 0;
+ if (rmdir(dir) < 0 && errno != ENOENT) {
+ ret = -errno;
+ goto out;
+ }
+
+ ret = 0;
+out:
```

[PATCH 2/2] uml: rename and improve actually_do_remove()

[PATCH 2/2] uml: rename and improve actually_do_remove()

```
+ closedir(directory);
+ return ret;
}

/* This says that there isn't already a user of the specified directory even if
@@ -172,9 +193,9 @@ static int umdir_take_if_dead(char *dir)
if (is_umdir_used(dir))
return -EEXIST;

- ret = actually_do_remove(dir);
+ ret = remove_files_and_dir(dir);
if (ret) {
- printk("is_umdir_used - actually_do_remove failed with "
+ printk("is_umdir_used - remove_files_and_dir failed with "
"err = %d\n", ret);
}
return ret;
@@ -354,9 +375,9 @@ static void remove_umid_dir(void)
char dir[strlen(uml_dir) + UMID_LEN + 1], err;

sprintf(dir, "%s%s", uml_dir, umid);
- err = actually_do_remove(dir);
+ err = remove_files_and_dir(dir);
if(err)
- printf("remove_umid_dir - actually_do_remove failed with "
+ printf("remove_umid_dir - remove_files_and_dir failed with "
"err = %d\n", err);
}

-
```

To unsubscribe from this list: send the line "unsubscribe linux-kernel" in the body of a message to majordomo@xxxxxxxxxxxxxxxxxxx
More majordomo info at <http://vger.kernel.org/majordomo-info.html>
Please read the FAQ at <http://www.tux.org/lkml/>