
Subject: [PATCH 7/12] allow proc_dir_entries to have destructor
Posted by [Mishin Dmitry](#) on Wed, 06 Dec 2006 22:27:41 GMT
[View Forum Message](#) <> [Reply to Message](#)

Destructor field added proc_dir_entries,
standard destructor kfree'ing data introduced.

Signed-off-by: Andrey Savochkin <saw@swsoft.com>

```
---
fs/proc/generic.c      | 10 ++++++---
fs/proc/root.c         |  1 +
include/linux/proc_fs.h |  4 ++++
3 files changed, 13 insertions(+), 2 deletions(-)

--- linux-2.6.19-rc6-mm2.orig/fs/proc/generic.c
+++ linux-2.6.19-rc6-mm2/fs/proc/generic.c
@@ -608,6 +608,11 @@ static struct proc_dir_entry *proc_creat
    return ent;
}

+void proc_data_destructor(struct proc_dir_entry *ent)
+{
+    kfree(ent->data);
+}
+
struct proc_dir_entry *proc_symlink(const char *name,
    struct proc_dir_entry *parent, const char *dest)
{
@@ -620,6 +625,7 @@ struct proc_dir_entry *proc_symlink(const
    ent->data = kmalloc((ent->size=strlen(dest))+1, GFP_KERNEL);
    if (ent->data) {
        strcpy((char*)ent->data, dest);
+    ent->destructor = proc_data_destructor;
    if (proc_register(parent, ent) < 0) {
        kfree(ent->data);
        kfree(ent);
@@ -698,8 +704,8 @@ void free_proc_entry(struct proc_dir_ent

    release_inode_number(ino);

- if (S_ISLNK(de->mode) && de->data)
- kfree(de->data);
+ if (de->destructor)
+ de->destructor(de);
    kfree(de);
}
```

```

--- linux-2.6.19-rc6-mm2.orig/fs/proc/root.c
+++ linux-2.6.19-rc6-mm2/fs/proc/root.c
@@ -166,6 +166,7 @@ EXPORT_SYMBOL(proc_symlink);
EXPORT_SYMBOL(proc_mkdir);
EXPORT_SYMBOL(create_proc_entry);
EXPORT_SYMBOL(remove_proc_entry);
+EXPORT_SYMBOL(proc_data_destructor);
EXPORT_SYMBOL(proc_root);
EXPORT_SYMBOL(proc_root_fs);
EXPORT_SYMBOL(proc_net);
--- linux-2.6.19-rc6-mm2.orig/include/linux/proc_fs.h
+++ linux-2.6.19-rc6-mm2/include/linux/proc_fs.h
@@ -45,6 +45,8 @@ typedef int (read_proc_t)(char *page, ch
typedef int (write_proc_t)(struct file *file, const char __user *buffer,
    unsigned long count, void *data);
typedef int (get_info_t)(char *, char **, off_t, int);
+struct proc_dir_entry;
+typedef void (destroy_proc_t)(struct proc_dir_entry *);

struct proc_dir_entry {
    unsigned int low_ino;
@@ -64,6 +66,7 @@ struct proc_dir_entry {
    read_proc_t *read_proc;
    write_proc_t *write_proc;
    atomic_t count; /* use count */
+ destroy_proc_t *destructor;
    int deleted; /* delete flag */
    void *set;
};
@@ -108,6 +111,7 @@ char *task_mem(struct mm_struct *, char
extern struct proc_dir_entry *create_proc_entry(const char *name, mode_t mode,
    struct proc_dir_entry *parent);
extern void remove_proc_entry(const char *name, struct proc_dir_entry *parent);
+extern void proc_data_destructor(struct proc_dir_entry *);

extern struct vfsmount *proc_mnt;
extern int proc_fill_super(struct super_block *, void *, int);

```

Containers mailing list
Containers@lists.osdl.org
<https://lists.osdl.org/mailman/listinfo/containers>
