Subject: [PATCH 6/10] Bsdacct: make the acct_lock global
Posted by Pavel Emelianov on Thu, 10 Apr 2008 08:28:58 GMT
View Forum Message <> Reply to Message

Don't use per-bsd-acct-struct lock, but work with a global
one. This lock is taken for short periods, so it's not going
to be a bottleneck, but it will allow us to easily avoid
many locking difficulties in the future.

Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

```
---
 kernel/acct.c |   42 +++++++++++++++++++++---------------------
 1 files changed, 21 insertions(+), 21 deletions(-)

diff --git a/kernel/acct.c b/kernel/acct.c
index 05f8bc0..fc71c13 100644
--- a/kernel/acct.c
+++ b/kernel/acct.c
@@ -83,7 +83,6 @@ static void do_acct_process(struct pid_namespace *ns, struct file *);
  * the cache line to have the data after getting the lock.
  */
 struct bsd_acct_struct {
- spinlock_t  lock;
  volatile int  active;
  volatile int  needcheck;
  struct file  *file;
@@ -91,8 +90,9 @@ struct bsd_acct_struct {
  struct timer_list timer;
 };

-static struct bsd_acct_struct acct_globals __cacheline_aligned =
- {__SPIN_LOCK_UNLOCKED(acct_globals.lock)};
+static DEFINE_SPINLOCK(acct_lock);
+
+static struct bsd_acct_struct acct_globals __cacheline_aligned;

 /*
  * Called whenever the timer says to check the free space.
@@ -114,11 +114,11 @@ static int check_free_space(struct file *file)
  sector_t resume;
  sector_t suspend;

- spin_lock(&acct_globals.lock);
+ spin_lock(&acct_lock);
  res = acct_globals.active;
  if (!file || !acct_globals.needcheck)
   goto out;
```

```
- spin_unlock(&acct_globals.lock);
+ spin_unlock(&acct_lock);

  /* May block */
  if (vfs_statfs(file->f_path.dentry, &sbuf))
@@ -140,7 +140,7 @@ static int check_free_space(struct file *file)
   * If some joker switched acct_globals.file under us we'ld better be
   * silent and _not_ touch anything.
   */
- spin_lock(&acct_globals.lock);
+ spin_lock(&acct_lock);
  if (file != acct_globals.file) {
   if (act)
    res = act>0;
@@ -165,7 +165,7 @@ static int check_free_space(struct file *file)
  add_timer(&acct_globals.timer);
  res = acct_globals.active;
 out:
- spin_unlock(&acct_globals.lock);
+ spin_unlock(&acct_lock);
  return res;
 }

@@ -173,7 +173,7 @@ out:
  * Close the old accounting file (if currently open) and then replace
  * it with file (if non-NULL).
  *
- * NOTE: acct_globals.lock MUST be held on entry and exit.
+ * NOTE: acct_lock MUST be held on entry and exit.
  */
 static void acct_file_reopen(struct file *file)
 {
@@ -201,11 +201,11 @@ static void acct_file_reopen(struct file *file)
  }
  if (old_acct) {
   mnt_unpin(old_acct->f_path.mnt);
-  spin_unlock(&acct_globals.lock);
+  spin_unlock(&acct_lock);
   do_acct_process(old_ns, old_acct);
   filp_close(old_acct, NULL);
   put_pid_ns(old_ns);
-  spin_lock(&acct_globals.lock);
+  spin_lock(&acct_lock);
  }
 }

@@ -235,10 +235,10 @@ static int acct_on(char *name)
   return error;
```

```
  }

- spin_lock(&acct_globals.lock);
+ spin_lock(&acct_lock);
  mnt_pin(file->f_path.mnt);
  acct_file_reopen(file);
- spin_unlock(&acct_globals.lock);
+ spin_unlock(&acct_lock);

  mntput(file->f_path.mnt); /* it's pinned, now give up active reference */

@@ -272,9 +272,9 @@ asmlinkage long sys_acct(const char __user *name)
  } else {
   error = security_acct(NULL);
   if (!error) {
-   spin_lock(&acct_globals.lock);
+   spin_lock(&acct_lock);
    acct_file_reopen(NULL);
-   spin_unlock(&acct_globals.lock);
+   spin_unlock(&acct_lock);
   }
  }
  return error;
@@ -289,10 +289,10 @@ asmlinkage long sys_acct(const char __user *name)
  */
 void acct_auto_close_mnt(struct vfsmount *m)
 {
- spin_lock(&acct_globals.lock);
+ spin_lock(&acct_lock);
  if (acct_globals.file && acct_globals.file->f_path.mnt == m)
   acct_file_reopen(NULL);
- spin_unlock(&acct_globals.lock);
+ spin_unlock(&acct_lock);
 }

 /**
@@ -304,12 +304,12 @@ void acct_auto_close_mnt(struct vfsmount *m)
  */
 void acct_auto_close(struct super_block *sb)
 {
- spin_lock(&acct_globals.lock);
+ spin_lock(&acct_lock);
  if (acct_globals.file &&
      acct_globals.file->f_path.mnt->mnt_sb == sb) {
   acct_file_reopen(NULL);
  }
- spin_unlock(&acct_globals.lock);
+ spin_unlock(&acct_lock);
```

```
 }

 /*
@@ -594,15 +594,15 @@ void acct_process(void)
  if (!acct_globals.file)
   return;

- spin_lock(&acct_globals.lock);
+ spin_lock(&acct_lock);
  file = acct_globals.file;
  if (unlikely(!file)) {
-  spin_unlock(&acct_globals.lock);
+  spin_unlock(&acct_lock);
   return;
  }
  get_file(file);
  ns = get_pid_ns(acct_globals.ns);
- spin_unlock(&acct_globals.lock);
+ spin_unlock(&acct_lock);

  do_acct_process(ns, file);
  fput(file);
--
1.5.3.4
```