
Subject: [PATCH 1/16] Round up the API

Posted by [Pavel Emelianov](#) on Fri, 06 Jul 2007 08:03:17 GMT

[View Forum Message](#) <> [Reply to Message](#)

The set of functions `process_session`, `task_session`, `process_group` and `task_pgrp` is confusing, as the names can be mixed with each other when looking at the code for a long time.

The proposals are to

- * equip the functions that return the integer with `_nr` suffix to represent that fact,
- * and to make all functions work with task (not process) by making the common prefix of the same name.

For monotony the routines `signal_session()` and `set_signal_session()` are replaced with `task_session_nr()` and `set_task_session()`, especially since they are only used with the explicit task->signal dereference.

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

Acked-by: Serge E. Hallyn <serue@us.ibm.com>

```
arch/mips/kernel/irixelf.c | 4 ++--
arch/mips/kernel/irixsig.c | 2 +-
arch/mips/kernel/sysirix.c | 4 ++--
arch/sparc64/solaris/misc.c | 4 ++--
drivers/char/tty_io.c      | 4 ++--
fs/autofs/inode.c          | 2 +-
fs/autofs/root.c           | 4 ++--
fs/autofs4/autofs_i.h      | 2 +-
fs/autofs4/inode.c         | 4 ++--
fs/autofs4/root.c          | 4 ++--
fs/binfmt_elf.c            | 8 ++++----
fs/binfmt_elf_fdpic.c     | 8 ++++----
fs/coda/upcall.c           | 2 +-
fs/proc/array.c            | 4 ++--
include/linux/sched.h      | 15 ++++-----
kernel/exit.c              | 10 ++++-----
kernel/fork.c              | 4 ++--
kernel/signal.c            | 2 +-
kernel/sys.c               | 14 ++++-----
19 files changed, 48 insertions(+), 53 deletions(-)
```

--- ./arch/mips/kernel/irixelf.c.apiren 2007-06-14 12:14:29.000000000 +0400

+++ ./arch/mips/kernel/irixelf.c 2007-06-14 15:52:54.000000000 +0400

```
@@ -1170,8 +1170,8 @@ static int irix_core_dump(long signr, st
    prstatus.pr_sighold = current->blocked.sig[0];
```

```

psinfo.pr_pid = prstatus.pr_pid = current->pid;
psinfo.pr_ppid = prstatus.pr_ppid = current->parent->pid;
- psinfo.pr_pgrp = prstatus.pr_pgrp = process_group(current);
+ psinfo.pr_pgrp = prstatus.pr_pgrp = task_pgrp_nr(current);
+ psinfo.pr_sid = prstatus.pr_sid = task_session_nr(current);
if (current->pid == current->tgid) {
/*
 * This is the record for the group leader. Add in the
--- ./arch/mips/kernel/irixsig.c.apiren 2007-06-14 12:14:29.000000000 +0400
+++ ./arch/mips/kernel/irixsig.c 2007-06-14 15:52:54.000000000 +0400
@@ -609,7 +609,7 @@ repeat:
    p = list_entry(_p,struct task_struct,sibling);
    if ((type == IRIX_P_PID) && p->pid != pid)
        continue;
- if ((type == IRIX_P_PGID) && process_group(p) != pid)
+ if ((type == IRIX_P_PGID) && task_pgrp_nr(p) != pid)
    continue;
    if ((p->exit_signal != SIGCHLD))
        continue;
--- ./arch/mips/kernel/sysirix.c.apiren 2007-06-14 12:14:29.000000000 +0400
+++ ./arch/mips/kernel/sysirix.c 2007-06-14 15:52:54.000000000 +0400
@@ -763,11 +763,11 @@ asmlinkage int irix_setpgrp(int flags)
    printk("[%s:%d] setpgrp(%d) ", current->comm, current->pid, flags);
#endif
    if(!flags)
- error = process_group(current);
+ error = task_pgrp_nr(current);
    else
        error = sys_setsid();
#ifdef DEBUG_PROCGRPS
- printk("returning %d\n", process_group(current));
+ printk("returning %d\n", task_pgrp_nr(current));
#endif

    return error;
--- ./arch/sparc64/solaris/misc.c.apiren 2007-06-14 12:14:29.000000000 +0400
+++ ./arch/sparc64/solaris/misc.c 2007-06-14 15:52:54.000000000 +0400
@@ -415,7 +415,7 @@ asmlinkage int solaris_procids(int cmd,

    switch (cmd) {
    case 0: /* getpgrp */
- return process_group(current);
+ return task_pgrp_nr(current);
    case 1: /* setpgrp */
    {
        int (*sys_setpgid)(pid_t,pid_t) =
@@ -426,7 +426,7 @@ asmlinkage int solaris_procids(int cmd,

```

```

    ret = sys_setpgid(0, 0);
    if (ret) return ret;
    proc_clear_tty(current);
-   return process_group(current);
+   return task_pgrp_nr(current);
}
case 2: /* getsid */
{
--- ./drivers/char/tty_io.c.apiren 2007-06-14 12:14:29.000000000 +0400
+++ ./drivers/char/tty_io.c 2007-06-14 15:52:55.000000000 +0400
@@ -3483,7 +3483,7 @@ void __do_SAK(struct tty_struct *tty)
/* Kill the entire session */
do_each_pid_task(session, PIDTYPE_SID, p) {
    printk(KERN_NOTICE "SAK: killed process %d"
-   " (%s): process_session(p)==tty->session\n",
+   " (%s): task_session_nr(p)==tty->session\n",
    p->pid, p->comm);
    send_sig(SIGKILL, p, 1);
} while_each_pid_task(session, PIDTYPE_SID, p);
@@ -3493,7 +3493,7 @@ void __do_SAK(struct tty_struct *tty)
do_each_thread(g, p) {
    if (p->signal->tty == tty) {
        printk(KERN_NOTICE "SAK: killed process %d"
-       " (%s): process_session(p)==tty->session\n",
+       " (%s): task_session_nr(p)==tty->session\n",
        p->pid, p->comm);
        send_sig(SIGKILL, p, 1);
        continue;
--- ./fs/autofs/inode.c.apiren 2007-06-14 12:14:29.000000000 +0400
+++ ./fs/autofs/inode.c 2007-06-14 15:52:55.000000000 +0400
@@ -80,7 +80,7 @@ static int parse_options(char *options,

*uid = current->uid;
*gid = current->gid;
- *pgrp = process_group(current);
+ *pgrp = task_pgrp_nr(current);

*minproto = *maxproto = AUTOFS_PROTO_VERSION;

--- ./fs/autofs/root.c.apiren 2007-06-14 12:14:29.000000000 +0400
+++ ./fs/autofs/root.c 2007-06-14 15:52:55.000000000 +0400
@@ -215,7 +215,7 @@ static struct dentry *autofs_root_lookup
    oz_mode = autofs_oz_mode(sbi);
    DPRINTK(("autofs_lookup: pid = %u, pgrp = %u, catatonic = %d, "
        "oz_mode = %d\n", pid_nr(task_pid(current)),
-   process_group(current), sbi->catatonic,
+   task_pgrp_nr(current), sbi->catatonic,
    oz_mode));

```

```

/*
@@ -536,7 +536,7 @@ static int autofs_root_ioctl(struct inode
    struct autofs_sb_info *sbi = autofs_sb_info(inode->i_sb);
    void __user *argp = (void __user *)arg;

- DPRINTK(("autofs_ioctl: cmd = 0x%08x, arg = 0x%08lx, sbi = %p, pgrp =
%u\n",cmd,arg,sbi,process_group(current)));
+ DPRINTK(("autofs_ioctl: cmd = 0x%08x, arg = 0x%08lx, sbi = %p, pgrp =
%u\n",cmd,arg,sbi,task_pgrp_nr(current)));

    if (_IOC_TYPE(cmd) != _IOC_TYPE(AUTOFS_IOC_FIRST) ||
        _IOC_NR(cmd) - _IOC_NR(AUTOFS_IOC_FIRST) >= AUTOFS_IOC_COUNT)
--- ./fs/autofs4/autofs_i.h.apiren 2007-06-14 12:14:29.000000000 +0400
+++ ./fs/autofs4/autofs_i.h 2007-06-14 15:52:55.000000000 +0400
@@ -131,7 +131,7 @@ static inline struct autofs_info *autofs
    filesystem without "magic".) */

static inline int autofs4_oz_mode(struct autofs_sb_info *sbi) {
- return sbi->catatonic || process_group(current) == sbi->oz_pgrp;
+ return sbi->catatonic || task_pgrp_nr(current) == sbi->oz_pgrp;
}

/* Does a dentry have some pending activity? */
--- ./fs/autofs4/inode.c.apiren 2007-06-14 12:14:29.000000000 +0400
+++ ./fs/autofs4/inode.c 2007-06-14 15:52:55.000000000 +0400
@@ -226,7 +226,7 @@ static int parse_options(char *options,

    *uid = current->uid;
    *gid = current->gid;
- *pgrp = process_group(current);
+ *pgrp = task_pgrp_nr(current);

    *minproto = AUTOFS_MIN_PROTO_VERSION;
    *maxproto = AUTOFS_MAX_PROTO_VERSION;
@@ -325,7 +325,7 @@ int autofs4_fill_super(struct super_bloc
    sbi->pipe = NULL;
    sbi->catatonic = 1;
    sbi->exp_timeout = 0;
- sbi->oz_pgrp = process_group(current);
+ sbi->oz_pgrp = task_pgrp_nr(current);
    sbi->sb = s;
    sbi->version = 0;
    sbi->sub_version = 0;
--- ./fs/autofs4/root.c.apiren 2007-06-14 12:14:29.000000000 +0400
+++ ./fs/autofs4/root.c 2007-06-14 15:52:55.000000000 +0400
@@ -582,7 +582,7 @@ static struct dentry *autofs4_lookup(str
    oz_mode = autofs4_oz_mode(sbi);

```

```

DPRINTK("pid = %u, pgrp = %u, catatonic = %d, oz_mode = %d",
- current->pid, process_group(current), sbi->catatonic, oz_mode);
+ current->pid, task_pgrp_nr(current), sbi->catatonic, oz_mode);

unhashed = autofs4_lookup_unhashed(sbi, dentry->d_parent, &dentry->d_name);
if (!unhashed) {
@@ -973,7 +973,7 @@ static int autofs4_root_ioctl(struct ino
void __user *p = (void __user *)arg;

DPRINTK("cmd = 0x%08x, arg = 0x%08lx, sbi = %p, pgrp = %u",
- cmd,arg,sbi,process_group(current));
+ cmd,arg,sbi,task_pgrp_nr(current));

if (_IOC_TYPE(cmd) != _IOC_TYPE(AUTOFS_IOC_FIRST) ||
    _IOC_NR(cmd) - _IOC_NR(AUTOFS_IOC_FIRST) >= AUTOFS_IOC_COUNT)
--- ./fs/binfmt_elf.c.apiren 2007-06-14 12:14:29.000000000 +0400
+++ ./fs/binfmt_elf.c 2007-06-14 15:52:55.000000000 +0400
@@ -1394,8 +1394,8 @@ static void fill_prstatus(struct elf_prs
prstatus->pr_sighold = p->blocked.sig[0];
prstatus->pr_pid = p->pid;
prstatus->pr_ppid = p->parent->pid;
- prstatus->pr_pgrp = process_group(p);
- prstatus->pr_sid = process_session(p);
+ prstatus->pr_pgrp = task_pgrp_nr(p);
+ prstatus->pr_sid = task_session_nr(p);
if (thread_group_leader(p)) {
/*
* This is the record for the group leader. Add in the
@@ -1440,8 +1440,8 @@ static int fill_psinfo(struct elf_prpsin

psinfo->pr_pid = p->pid;
psinfo->pr_ppid = p->parent->pid;
- psinfo->pr_pgrp = process_group(p);
- psinfo->pr_sid = process_session(p);
+ psinfo->pr_pgrp = task_pgrp_nr(p);
+ psinfo->pr_sid = task_session_nr(p);

i = p->state ? ffz(~p->state) + 1 : 0;
psinfo->pr_state = i;
--- ./fs/binfmt_elf_fdpic.c.apiren 2007-06-14 12:14:29.000000000 +0400
+++ ./fs/binfmt_elf_fdpic.c 2007-06-14 15:52:55.000000000 +0400
@@ -1344,8 +1344,8 @@ static void fill_prstatus(struct elf_prs
prstatus->pr_sighold = p->blocked.sig[0];
prstatus->pr_pid = p->pid;
prstatus->pr_ppid = p->parent->pid;
- prstatus->pr_pgrp = process_group(p);
- prstatus->pr_sid = process_session(p);

```

```

+ prstatus->pr_pgrp = task_pgrp_nr(p);
+ prstatus->pr_sid = task_session_nr(p);
  if (thread_group_leader(p)) {
/*
  * This is the record for the group leader. Add in the
@@ -1393,8 +1393,8 @@ static int fill_psinfo(struct elf_prpsin

  psinfo->pr_pid = p->pid;
  psinfo->pr_ppid = p->parent->pid;
- psinfo->pr_pgrp = process_group(p);
- psinfo->pr_sid = process_session(p);
+ psinfo->pr_pgrp = task_pgrp_nr(p);
+ psinfo->pr_sid = task_session_nr(p);

  i = p->state ? ffz(~p->state) + 1 : 0;
  psinfo->pr_state = i;
--- ./fs/coda/upcall.c.apiren 2007-06-14 12:14:29.000000000 +0400
+++ ./fs/coda/upcall.c 2007-06-14 15:52:55.000000000 +0400
@@ -53,7 +53,7 @@ static void *alloc_upcall(int opcode, in

  inp->ih.opcode = opcode;
  inp->ih.pid = current->pid;
- inp->ih.pgid = process_group(current);
+ inp->ih.pgid = task_pgrp_nr(current);
#ifdef CONFIG_CODA_FS_OLD_API
  memset(&inp->ih.cred, 0, sizeof(struct coda_cred));
  inp->ih.cred.cr_fsuid = current->fsuid;
--- ./fs/proc/array.c.apiren 2007-06-14 12:14:29.000000000 +0400
+++ ./fs/proc/array.c 2007-06-14 15:52:55.000000000 +0400
@@ -414,8 +414,8 @@ static int do_task_stat(struct task_stru
  stime += cputime_to_clock_t(sig->stime);
}

- sid = signal_session(sig);
- pgid = process_group(task);
+ sid = task_session_nr(task);
+ pgid = task_pgrp_nr(task);
  ppid = rcu_dereference(task->real_parent)->tgid;

  unlock_task_sighand(task, &flags);
--- ./include/linux/sched.h.apiren 2007-06-14 12:14:29.000000000 +0400
+++ ./include/linux/sched.h 2007-06-14 15:52:55.000000000 +0400
@@ -1153,24 +1153,19 @@ struct task_struct {
#endif
};

-static inline pid_t process_group(struct task_struct *tsk)
+static inline pid_t task_pgrp_nr(struct task_struct *tsk)

```

```

{
    return tsk->signal->pgrp;
}

-static inline pid_t signal_session(struct signal_struct *sig)
-{
- return sig->__session;
-}
-
-static inline pid_t process_session(struct task_struct *tsk)
+static inline pid_t task_session_nr(struct task_struct *tsk)
{
- return signal_session(tsk->signal);
+ return tsk->signal->__session;
}

-static inline void set_signal_session(struct signal_struct *sig, pid_t session)
+static inline void set_task_session(struct task_struct *tsk, pid_t session)
{
- sig->__session = session;
+ tsk->signal->__session = session;
}

static inline struct pid *task_pid(struct task_struct *task)
--- ./kernel/exit.c.apiren 2007-06-14 12:14:29.000000000 +0400
+++ ./kernel/exit.c 2007-06-14 15:52:55.000000000 +0400
@@ -309,12 +309,12 @@ void __set_special_pids(pid_t session, p
{
    struct task_struct *curr = current->group_leader;

- if (process_session(curr) != session) {
+ if (task_session_nr(curr) != session) {
    detach_pid(curr, PIDTYPE_SID);
- set_signal_session(curr->signal, session);
+ set_task_session(curr, session);
    attach_pid(curr, PIDTYPE_SID, find_pid(session));
}
- if (process_group(curr) != pgrp) {
+ if (task_pgrp_nr(curr) != pgrp) {
    detach_pid(curr, PIDTYPE_PGID);
    curr->signal->pgrp = pgrp;
    attach_pid(curr, PIDTYPE_PGID, find_pid(pgrp));
@@ -1055,10 +1055,10 @@ static int eligible_child(pid_t pid, int
    if (p->pid != pid)
        return 0;
} else if (!pid) {
- if (process_group(p) != process_group(current))
+ if (task_pgrp_nr(p) != task_pgrp_nr(current))

```

```

    return 0;
} else if (pid != -1) {
- if (process_group(p) != -pid)
+ if (task_pgrp_nr(p) != -pid)
    return 0;
}

```

```

--- ./kernel/fork.c.apiren 2007-06-14 12:14:29.000000000 +0400
+++ ./kernel/fork.c 2007-06-14 15:52:55.000000000 +0400
@@ -1259,8 +1259,8 @@ static struct task_struct *copy_process(

```

```

    if (thread_group_leader(p)) {
        p->signal->tty = current->signal->tty;
-   p->signal->pgrp = process_group(current);
-   set_signal_session(p->signal, process_session(current));
+   p->signal->pgrp = task_pgrp_nr(current);
+   set_task_session(p, task_session_nr(current));
        attach_pid(p, PIDTYPE_PGID, task_pgrp(current));
        attach_pid(p, PIDTYPE_SID, task_session(current));

```

```

--- ./kernel/signal.c.apiren 2007-06-14 12:14:29.000000000 +0400
+++ ./kernel/signal.c 2007-06-14 15:52:55.000000000 +0400
@@ -517,7 +517,7 @@ static int check_kill_permission(int sig
    error = -EPERM;
    if ((info == SEND_SIG_NOINFO || (!is_si_special(info) && SI_FROMUSER(info)))
        && ((sig != SIGCONT) ||
-   (process_session(current) != process_session(t)))
+   (task_session_nr(current) != task_session_nr(t)))
        && (current->euid ^ t->suid) && (current->euid ^ t->uid)
        && (current->uid ^ t->suid) && (current->uid ^ t->uid)
        && !capable(CAP_KILL))

```

```

--- ./kernel/sys.c.apiren 2007-06-14 12:14:29.000000000 +0400
+++ ./kernel/sys.c 2007-06-14 15:52:55.000000000 +0400
@@ -1485,7 +1485,7 @@ asmlinkage long sys_setpgid(pid_t pid, p
    if (err)
        goto out;

```

```

- if (process_group(p) != pgid) {
+ if (task_pgrp_nr(p) != pgid) {
    detach_pid(p, PIDTYPE_PGID);
    p->signal->pgrp = pgid;
    attach_pid(p, PIDTYPE_PGID, find_pid(pgid));
@@ -1501,7 +1501,7 @@ out:
asmlinkage long sys_getpgid(pid_t pid)
{
    if (!pid)
-   return process_group(current);
+   return task_pgrp_nr(current);

```

```

else {
    int retval;
    struct task_struct *p;
@@ -1513,7 +1513,7 @@ asmlinkage long sys_getpgid(pid_t pid)
    if (p) {
        retval = security_task_getpgid(p);
        if (!retval)
-       retval = process_group(p);
+       retval = task_pgrp_nr(p);
    }
    read_unlock(&tasklist_lock);
    return retval;
@@ -1525,7 +1525,7 @@ asmlinkage long sys_getpgid(pid_t pid)
asmlinkage long sys_getpgrp(void)
{
    /* SMP - assuming writes are word atomic this is fine */
-   return process_group(current);
+   return task_pgrp_nr(current);
}

#endif
@@ -1533,7 +1533,7 @@ asmlinkage long sys_getpgrp(void)
asmlinkage long sys_getsid(pid_t pid)
{
    if (!pid)
-   return process_session(current);
+   return task_session_nr(current);
    else {
        int retval;
        struct task_struct *p;
@@ -1545,7 +1545,7 @@ asmlinkage long sys_getsid(pid_t pid)
    if (p) {
        retval = security_task_getsid(p);
        if (!retval)
-       retval = process_session(p);
+       retval = task_session_nr(p);
    }
    read_unlock(&tasklist_lock);
    return retval;
@@ -1582,7 +1582,7 @@ asmlinkage long sys_setsid(void)
    group_leader->signal->tty = NULL;
    spin_unlock(&group_leader->sigband->siglock);

-   err = process_group(group_leader);
+   err = task_pgrp_nr(group_leader);
out:
    write_unlock_irq(&tasklist_lock);
    return err;

```

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