Subject: Re: [RFC][PATCH] Do not set /proc inode->pid for non-pid-related inodes
Posted by Cedric Le Goater on Wed, 21 Mar 2007 17:29:35 GMT
View Forum Message <> Reply to Message

>> what about a kthread that would be spawned when a task is cloned in an
>> unshared pid namespace ? This is an extra cost in term of tasks.
>
> If you use kernel_thread this can happen. (Die kernel_thread).
> If you use the kthread interface keventd will be the parent process and
> we won't have problems.

so is it something acceptable for mainline ? I think openvz has such
a thread doing the reaping.

> Thus most users of kernel_thread need to be fixed to use the kthread
> interface.
>
> Thanks for the reminder of this one, I had forgotten that bit of
> reasoning for updating kernel_thread users.

there are not much left. see below a quick and dirty survey on
2.6.21-rc4-mm1.

C.

```
./fs/jffs2/background.c: pid = kernel_thread(jffs2_garbage_collect_thread, c,
CLONE_FS|CLONE_FILES);
./fs/nfs/delegation.c: status = kernel_thread(recall_thread, &data, CLONE_KERNEL);
./fs/cifs/connect.c:   rc = (int)kernel_thread((void *)(void *)cifs_demultiplex_thread, srvTcp,
./fs/lockd/clntlock.c:  if (kernel_thread(reclaimer, host, CLONE_KERNEL) < 0)
./fs/afs/kafsasyncd.c: ret = kernel_thread(kafsasyncd, NULL, 0);
./fs/afs/kafstimod.c: ret = kernel_thread(kafstimod, NULL, 0);
./fs/afs/cmservice.c:  ret = kernel_thread(kafscmd, NULL, 0);
./arch/powerpc/platforms/pseries/rtasd.c: if (kernel_thread(rtasd, NULL, CLONE_FS) < 0)
./arch/powerpc/platforms/pseries/eeh_event.c: if (kernel_thread(eeh_event_handler, NULL,
CLONE_KERNEL) < 0)
./arch/ia64/sn/kernel/xpc_main.c: pid = kernel_thread(xpc_activating, (void *) ((u64) partid), 0);
./arch/ia64/sn/kernel/xpc_main.c:  pid = kernel_thread(xpc_daemonize_kthread, (void *) args, 0);
./arch/ia64/sn/kernel/xpc_main.c: pid = kernel_thread(xpc_hb_checker, NULL, 0);
./arch/ia64/sn/kernel/xpc_main.c: pid = kernel_thread(xpc_initiate_discovery, NULL, 0);
./arch/arm/kernel/ecard.c: ret = kernel_thread(ecard_task, NULL, CLONE_KERNEL);
./arch/sparc64/kernel/power.c:  if (kernel_thread(powerd, NULL, CLONE_FS) < 0) {
./arch/i386/mach-voyager/voyager_thread.c: if(kernel_thread(thread, NULL, CLONE_KERNEL) <
0) {
./arch/i386/kernel/io_apic.c: if (kernel_thread(balanced_irq, NULL, CLONE_KERNEL) >= 0)
./arch/parisc/kernel/process.c: return __kernel_thread(fn, arg, flags);
./init/do_mounts_initrd.c: pid = kernel_thread(do_linuxrc, "/linuxrc", SIGCHLD);
```

./kernel/kmod.c: pid = kernel_thread(____call_usermodehelper, sub_info, SIGCHLD);
./kernel/kmod.c: pid = kernel_thread(wait_for_helper, sub_info,
./kernel/kmod.c: pid = kernel_thread(____call_usermodehelper, sub_info,
./kernel/stop_machine.c: ret = kernel_thread(stopmachine, (void *)(long)i,CLONE_KERNEL);
./net/ipv4/ipvs/ip_vs_sync.c: if ((pid = kernel_thread(sync_thread, startup, 0)) < 0) {
./net/ipv4/ipvs/ip_vs_sync.c: if ((pid = kernel_thread(fork_sync_thread, &startup, 0)) < 0) {
./net/sunrpc/svc.c: error = kernel_thread((int (*)(void *)) func, rqstp, 0);
./net/rxrpc/krxiod.c: return kernel_thread(rxrpc_krxiod, NULL, 0);
./net/rxrpc/krxsecd.c: return kernel_thread(rxrpc_krxsecd, NULL, 0);
./net/rxrpc/krxtimod.c: ret = kernel_thread(krxtimod, NULL, 0);
./net/bluetooth/bnep/core.c: err = kernel_thread(bnep_session, s, CLONE_KERNEL);
./net/bluetooth/hidp/core.c: err = kernel_thread(hidp_session, session, CLONE_KERNEL);
./net/bluetooth/cmtp/core.c: err = kernel_thread(cmtp_session, session, CLONE_KERNEL);
./net/bluetooth/rfcomm/core.c: kernel_thread(rfcomm_run, NULL, CLONE_KERNEL);
./drivers/media/video/saa7134/saa7134-tvaudio.c: dev->thread.pid =
kernel_thread(my_thread,dev,0);
./drivers/media/video/saa7134/saa7134-tvaudio.c: printk(KERN_WARNING "%s: kernel_thread()
failed\n",
./drivers/media/dvb/dvb-core/dvb_ca_en50221.c: ret = kernel_thread(dvb_ca_en50221_thread,
ca, 0);
./drivers/usb/atm/usbatm.c: int ret = kernel_thread(usbatm_do_heavy_init, instance,
CLONE_KERNEL);
./drivers/s390/scsi/zfcp_erp.c: retval = kernel_thread(zfcp_erp_thread, adapter, SIGCHLD);
./drivers/s390/net/lcs.c: kernel_thread(lcs_recovery, (void *) card, SIGCHLD);
./drivers/s390/net/lcs.c: kernel_thread(lcs_register_mc_addresses,
./drivers/s390/net/qeth_main.c: kernel_thread(qeth_recover, (void *) card, SIGCHLD);
./drivers/scsi/libsas/sas_scsi_host.c: res = kernel_thread(sas_queue_thread, sas_ha, 0);
./drivers/pnp/pnpbios/core.c: if (kernel_thread(pnp_dock_thread, NULL, CLONE_KERNEL) > 0)
./drivers/mtd/ubi/background.c: pid = kernel_thread(ubi_thread, ubi, CLONE_FS |
CLONE_FILES);
./drivers/mtd/mtd_blkdevs.c: ret = kernel_thread(mtd_blktrans_thread, tr, CLONE_KERNEL);
./drivers/pci/hotplug/cpci_hotplug_core.c: pid = kernel_thread(event_thread, NULL, 0);
./drivers/pci/hotplug/cpci_hotplug_core.c: pid = kernel_thread(poll_thread, NULL, 0);
./drivers/pci/hotplug/cpqphp_ctrl.c: pid = kernel_thread(event_thread, NULL, 0);
./drivers/macintosh/therm_windtunnel.c: x.poll_task = kernel_thread(control_loop, NULL,
SIGCHLD | CLONE_KERNEL);
./drivers/macintosh/mediabay.c: kernel_thread(media_bay_task, NULL, CLONE_KERNEL);
./drivers/macintosh/adb.c: adb_probe_task_pid = kernel_thread(adb_probe_task, NULL,
SIGCHLD | CLONE_KERNEL);
./drivers/macintosh/therm_pm72.c: ctrl_task = kernel_thread(main_control_loop, NULL, SIGCHLD
| CLONE_KERNEL);

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