
Subject: [PATCH -mm] utrace: fix double free re __rcu_process_callbacks()

Posted by [Alexey Dobriyan](#) on Tue, 24 Apr 2007 09:02:15 GMT

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

The following patch fixes double free manifesting itself as crash in

`__rcu_process_callbacks()`:

<http://marc.info/?l=linux-kernel&m=117518764517017&w=2>

https://bugzilla.redhat.com/bugzilla/show_bug.cgi?id=229112

The problem is with `check_dead_utrace()` conditionally scheduling "struct utrace" for freeing but not cleaning `struct task_struct::utrace` pointer leaving it reachable:

```
tsk->utrace_flags = flags;
if (flags)
    spin_unlock(&utrace->lock);
else
    rcu_utrace_free(utrace);
```

OTOH, `utrace_release_task()` first clears `->utrace` pointer, then frees struct utrace itself:

Roland inserted some debugging into 2.6.21-rc6-mm1 so that aforementioned double free couldn't be reproduced without seeing BUG at `kernel/utrace.c:176` first. It triggers if one struct utrace were passed to `rcu_utrace_free()` second time.

2-way P3, 8-way ia64, Core 2 Duo boxes. Testcase is at the first link.

I think it adds leak if `utrace_reap()` takes branch without freeing but, well, I hope Roland will give me some clue on how to fix it too.

Signed-off-by: Alexey Dobriyan <adobriyan@sw.ru>

kernel/utrace.c | 6 +-----

1 file changed, 1 insertion(+), 5 deletions(-)

but weren't easily reproducible without hitting double-free first.

FWIW, it's

```
BUG_ON(!list_empty(&tsk->ptracees));
oops at the beginning of remove_engine()
NULL ->report_quiesce call which is absent in ptrace utrace ops
BUG_ON(tracehook_check_released(p));
```

--- a/kernel/utrace.c

```

+++ b/kernel/utrace.c
@@ -205,7 +205,6 @@ utrace_clear_tsk(struct task_struct *tsk
    if (utrace->u.live.signal == NULL) {
        task_lock(tsk);
        if (likely(tsk->utrace != NULL)) {
-   rcu_assign_pointer(tsk->utrace, NULL);
        tsk->utrace_flags &= UTRACE_ACTION_NOEAP;
        }
        task_unlock(tsk);
@@ -305,10 +304,7 @@ check_dead_utrace(struct task_struct *tsk
    }

    tsk->utrace_flags = flags;
-   if (flags)
-       spin_unlock(&utrace->lock);
-   else
-       rcu_utrace_free(utrace);
+   spin_unlock(&utrace->lock);

    /*
     * Now we're finished updating the utrace state.

```
