Subject: [PATCH] add_timer -> mod_timer() in dst_run_gc()
Posted by Kirill Korotaev on Wed, 09 Aug 2006 09:00:26 GMT
View Forum Message <> Reply to Message

Patch from Dmitry Mishin <dim@openvz.org>:
Replace add_timer() by mod_timer() in dst_run_gc
in order to avoid BUG message.

```
      CPU1                        CPU2
dst_run_gc()  entered          dst_run_gc() entered
spin_lock(&dst_lock)                .....
del_timer(&dst_gc_timer)        fail to get lock
      ....                    mod_timer() <--- puts
                                  timer back
                                  to the list
add_timer(&dst_gc_timer) <--- BUG because timer is in list already.
```

Found during OpenVZ internal testing.

At first we thought that it is OpenVZ specific as we
added dst_run_gc(0) call in dst_dev_event(),
but as Alexey pointed to me it is possible to trigger
this condition in mainstream kernel.

F.e. timer has fired on CPU2, but the handler was preeempted
by an irq before dst_lock is tried.
Meanwhile, someone on CPU1 adds an entry to gc list and
starts the timer.
If CPU2 was preempted long enough, this timer can expire
simultaneously with resuming timer handler on CPU1, arriving
exactly to the situation described.

Signed-Off-By: Dmitry Mishin <dim@openvz.org>
Signed-Off-By: Kirill Korotaev <dev@openvz.org>
Signed-Off-By: Alexey Kuznetsov <kuznet@ms2.inr.ac.ru>

```
--- ./net/core/dst.c.dst 2006-05-19 13:12:34.000000000 +0400
+++ ./net/core/dst.c 2006-05-22 14:29:50.000000000 +0400
@@ -95,12 +95,11 @@ static void dst_run_gc(unsigned long dum
   dst_gc_timer_inc = DST_GC_INC;
   dst_gc_timer_expires = DST_GC_MIN;
  }
- dst_gc_timer.expires = jiffies + dst_gc_timer_expires;
 #if RT_CACHE_DEBUG >= 2
  printk("dst_total: %d/%d %ld\n",
         atomic_read(&dst_total), delayed,  dst_gc_timer_expires);
 #endif
```

```
- add_timer(&dst_gc_timer);
+ mod_timer(&dst_gc_timer, jiffies + dst_gc_timer_expires);

 out:
  spin_unlock(&dst_lock);
```