Subject: [PATCH] e1000: ring buffers resources cleanup
Posted by vaverin on Fri, 18 Aug 2006 15:02:34 GMT
View Forum Message <> Reply to Message

Memory leak was found in 2.6.18-rc4 and e1000 7.2.7 from sourceforge:
We should free resources allocated for previous rings if following allocation fails.

Signed-off-by: Vasily Averin <vvs@sw.ru>

Thank you,
Vasily Averin
SWsoft Virtuozzo/OpenVZ Linux kernel team

```
--- linux-2.6.18-rc4/drivers/net/e1000/e1000_main.c.irsrs 2006-08-18 16:58:51.000000000 +0400
+++ linux-2.6.18-rc4/drivers/net/e1000/e1000_main.c 2006-08-18 18:53:05.000000000 +0400
@@ -1398,6 +1398,9 @@ e1000_setup_all_tx_resources(struct e100
   if (err) {
    DPRINTK(PROBE, ERR,
     "Allocation for Tx Queue %u failed\n", i);
+   for (i-- ; i >= 0; i--)
+    e1000_free_tx_resources(adapter,
+       &adapter->tx_ring[i]);
    break;
   }
  }
@@ -1656,6 +1659,9 @@ e1000_setup_all_rx_resources(struct e100
   if (err) {
    DPRINTK(PROBE, ERR,
     "Allocation for Rx Queue %u failed\n", i);
+   for (i-- ; i >= 0; i--)
+    e1000_free_rx_resources(adapter,
+       &adapter->rx_ring[i]);
    break;
   }
  }
```

Subject: Re: [PATCH] e1000: ring buffers resources cleanup
Posted by vaverin on Sat, 19 Aug 2006 20:14:09 GMT
View Forum Message <> Reply to Message

Hello Joe,

Joe Perches wrote:
> On Fri, 2006-08-18 at 19:02 +0400, Vasily Averin wrote:
>>Memory leak was found in 2.6.18-rc4 and e1000 7.2.7 from sourceforge:
>>We should free resources allocated for previous rings if following allocation fails.

>
> Did you read the comment headers in the function?
>
> * If this function returns with an error, then it's possible one or
> * more of the rings is populated (while the rest are not). It is the
> * callers duty to clean those orphaned rings.

Thank you for your notice.
I believe this comment is incorrect: if some function returns an error it should
restore original state on exit, otherwise can lead to resource leaks. Also I
would note that this requirements is not accomplished in current driver version:
e1000_setup_all_Xx_resources functions are called in two places: in
e1000_set_ringparam() and in e1000_open() and in both cases nobody cleans those
orphaned rings.

Therefore I think it make sense to remove these comments too.

Andrew, could you please use attached patch instead previous version?

Memory leak was found in 2.6.18-rc4 and e1000 7.2.7 from sourceforge:
We should free resources allocated for previous rings if following allocation
fails. Also incorrect comments in e1000_setup_all_Xx_resources() are removed

Signed-off-by: Vasily Averin <vvs@sw.ru>

Thank you,
 Vasily Averin
SWsoft Virtuozzo/OpenVZ Linux kernel team

```
--- linux-2.6.18-rc4/drivers/net/e1000/e1000_main.c.irsrs 2006-08-18 16:58:51.000000000 +0400
+++ linux-2.6.18-rc4/drivers/net/e1000/e1000_main.c 2006-08-19 22:54:00.000000000 +0400
@@ -1381,10 +1381,6 @@ setup_tx_desc_die:
  *       (Descriptors) for all queues
  * @adapter: board private structure
  *
- * If this function returns with an error, then it's possible one or
- * more of the rings is populated (while the rest are not).  It is the
- * callers duty to clean those orphaned rings.
- *
  * Return 0 on success, negative on failure
  **/

@@ -1398,6 +1394,9 @@ e1000_setup_all_tx_resources(struct e100
   if (err) {
    DPRINTK(PROBE, ERR,
     "Allocation for Tx Queue %u failed\n", i);
+   for (i-- ; i >= 0; i--)
+    e1000_free_tx_resources(adapter,
```

```
+       &adapter->tx_ring[i]);
    break;
   }
  }
@@ -1639,10 +1638,6 @@ setup_rx_desc_die:
  *       (Descriptors) for all queues
  * @adapter: board private structure
  *
- * If this function returns with an error, then it's possible one or
- * more of the rings is populated (while the rest are not).  It is the
- * callers duty to clean those orphaned rings.
- *
  * Return 0 on success, negative on failure
  **/

@@ -1656,6 +1651,9 @@ e1000_setup_all_rx_resources(struct e100
   if (err) {
    DPRINTK(PROBE, ERR,
     "Allocation for Rx Queue %u failed\n", i);
+   for (i-- ; i >= 0; i--)
+    e1000_free_rx_resources(adapter,
+       &adapter->rx_ring[i]);
    break;
   }
  }
```