Subject: [PATCH] e1000: memory leak in e1000_set_ringparam()
Posted by vaverin on Fri, 18 Aug 2006 13:33:11 GMT
View Forum Message <> Reply to Message

Memory leak was found in 2.6.18-rc4 and e1000 7.2.7 from sourceforge:
Memory allocated for new tx_ring and rx_ring leaks if e1000_setup_XX_resources()
fails.c Also this patch reduces stack usage (removed tx_new and rx_new) and uses
kzalloc instead kmalloc+memset(0)

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_ethtool.c.igml 2006-08-18 16:58:51.000000000
+0400
+++ linux-2.6.18-rc4/drivers/net/e1000/e1000_ethtool.c 2006-08-18 16:59:53.000000000 +0400
@@ -619,8 +619,8 @@ e1000_set_ringparam(struct net_device *n
 {
  struct e1000_adapter *adapter = netdev_priv(netdev);
  e1000_mac_type mac_type = adapter->hw.mac_type;
- struct e1000_tx_ring *txdr, *tx_old, *tx_new;
- struct e1000_rx_ring *rxdr, *rx_old, *rx_new;
+ struct e1000_tx_ring *txdr, *tx_old;
+ struct e1000_rx_ring *rxdr, *rx_old;
  int i, err, tx_ring_size, rx_ring_size;

  if ((ring->rx_mini_pending) || (ring->rx_jumbo_pending))
@@ -638,23 +638,17 @@ e1000_set_ringparam(struct net_device *n
  tx_old = adapter->tx_ring;
  rx_old = adapter->rx_ring;

- adapter->tx_ring = kmalloc(tx_ring_size, GFP_KERNEL);
- if (!adapter->tx_ring) {
-  err = -ENOMEM;
-  goto err_setup_rx;
- }
- memset(adapter->tx_ring, 0, tx_ring_size);
-
- adapter->rx_ring = kmalloc(rx_ring_size, GFP_KERNEL);
- if (!adapter->rx_ring) {
-  kfree(adapter->tx_ring);
-  err = -ENOMEM;
-  goto err_setup_rx;
- }
- memset(adapter->rx_ring, 0, rx_ring_size);
+ err = -ENOMEM;
```

```
+ txdr = kzalloc(tx_ring_size, GFP_KERNEL);
+ if (!txdr)
+  goto err_alloc_tx;
+
+ rxdr = kzalloc(rx_ring_size, GFP_KERNEL);
+ if (!rxdr)
+  goto err_alloc_rx;

- txdr = adapter->tx_ring;
- rxdr = adapter->rx_ring;
+ adapter->tx_ring = txdr;
+ adapter->rx_ring = rxdr;

  rxdr->count = max(ring->rx_pending,(uint32_t)E1000_MIN_RXD);
  rxdr->count = min(rxdr->count,(uint32_t)(mac_type < e1000_82544 ?
@@ -681,16 +675,14 @@ e1000_set_ringparam(struct net_device *n
   /* save the new, restore the old in order to free it,
    * then restore the new back again */

-  rx_new = adapter->rx_ring;
-  tx_new = adapter->tx_ring;
   adapter->rx_ring = rx_old;
   adapter->tx_ring = tx_old;
   e1000_free_all_rx_resources(adapter);
   e1000_free_all_tx_resources(adapter);
   kfree(tx_old);
   kfree(rx_old);
-  adapter->rx_ring = rx_new;
-  adapter->tx_ring = tx_new;
+  adapter->rx_ring = rxdr;
+  adapter->tx_ring = txdr;
   if ((err = e1000_up(adapter)))
    goto err_setup;
  }
@@ -703,6 +695,10 @@ err_setup_tx:
 err_setup_rx:
  adapter->rx_ring = rx_old;
  adapter->tx_ring = tx_old;
+ kfree(rxdr);
+err_alloc_rx:
+ kfree(txdr);
+err_alloc_tx:
  e1000_up(adapter);
 err_setup:
  clear_bit(__E1000_RESETTING, &adapter->flags);
```