author	Ratheesh Kannoth <rkannoth@marvell.com>	2022-11-07 09:05:05 +0530
committer	Paolo Abeni <pabeni@redhat.com>	2022-11-08 13:43:46 +0100
commit	f0dfc4c88ef39be0ba736aa0ce6119263fc19aeb (patch)	
tree	cd03f12f8d4d0b93374d9529b173bea1e8e1fd8b	
parent	b0c09c7f08c2467b2089bdf4adb2fbbc2464f4a8 (diff)	
download	linux-f0dfc4c88ef39be0ba736aa0ce6119263fc19aeb.tar.gz	

diff options

context: 3

space: include

mode: unified

octeontx2-pf: Fix SQE threshold checking

```
Current way of checking available SQE count which is based on
HW updated SQB count could result in driver submitting an SQE
even before CQE for the previously transmitted SQE at the same
index is processed in NAPI resulting losing SKB pointers,
hence a leak. Fix this by checking a consumer index which
is updated once CQE is processed.

Fixes: 3ca6c4c882a7 ("octeontx2-pf: Add packet transmission support")
Signed-off-by: Ratheesh Kannoth <rkannoth@marvell.com>
Reviewed-by: Sunil Kovvuri Goutham <sgoutham@marvell.com>
Link: https://lore.kernel.org/r/20221107033505.2491464-1-rkannoth@marvell.com
Signed-off-by: Paolo Abeni <pabeni@redhat.com>
```

Diffstat

-rw-r--r--	drivers/net/ethernet/marvell/octeontx2/nic/otx2_common.c	1
-rw-r--r--	drivers/net/ethernet/marvell/octeontx2/nic/otx2_txrx.c	32
-rw-r--r--	drivers/net/ethernet/marvell/octeontx2/nic/otx2_txrx.h	1

3 files changed, 21 insertions, 13 deletions

```
diff --git a/drivers/net/ethernet/marvell/octeontx2/nic/otx2_common.c b/drivers/net/ethernet/marvell/octeontx2/nic/otx2_common.c
index 9ac9e6615ae717..9e10e7471b8874 100644
--- a/drivers/net/ethernet/marvell/octeontx2/nic/otx2_common.c
+++ b/drivers/net/ethernet/marvell/octeontx2/nic/otx2_common.c
@@ -898,6 +898,7 @@ static int otx2_sq_init(struct otx2_nic *pfvf, u16 qidx, u16 sqb_aura)
        }

        sq->head = 0;
+       sq->cons_head = 0;
        sq->sqe_per_sqb = (pfvf->hw.sqb_size / sq->sqe_size) - 1;
        sq->num_sqbs = (qset->sqe_cnt + sq->sqe_per_sqb) / sq->sqe_per_sqb;
        /* Set SQE threshold to 10% of total SQEs */

diff --git a/drivers/net/ethernet/marvell/octeontx2/nic/otx2_txrx.c b/drivers/net/ethernet/marvell/octeontx2/nic/otx2_txrx.c
index 5ec11d71bf6066..ef10aef3cda02d 100644
--- a/drivers/net/ethernet/marvell/octeontx2/nic/otx2_txrx.c
+++ b/drivers/net/ethernet/marvell/octeontx2/nic/otx2_txrx.c
@@ -441,6 +441,7 @@ static int otx2_tx_napi_handler(struct otx2_nic *pfvf,
                                struct otx2_cq_queue *cq, int budget)
 {
        int tx_pkts = 0, tx_bytes = 0, qidx;
+       struct otx2_snd_queue *sq;
        struct nix_cqe_tx_s *cqe;
        int processed_cqe = 0;

@@ -451,6 +452,9 @@ static int otx2_tx_napi_handler(struct otx2_nic *pfvf,
                return 0;

 process_cqe:
+       qidx = cq->cq_idx - pfvf->hw.rx_queues;
+       sq = &pfvf->qset.sq[qidx];
+
        while (likely(processed_cqe < budget) && cq->pend_cqe) {
                cqe = (struct nix_cqe_tx_s *)otx2_get_next_cqe(cq);
                if (unlikely(!cqe)) {
@@ -458,18 +462,20 @@ process_cqe:
                                return 0;
                        break;
                }
+
                if (cq->cq_type == CQ_XDP) {
```

```
-                       qidx = cq->cq_idx - pfvf->hw.rx_queues;
-                       otx2_xdp_snd_pkt_handler(pfvf, &pfvf->qset.sq[qidx],
-                                                cqe);
+                       otx2_xdp_snd_pkt_handler(pfvf, sq, cqe);
                } else {
-                       otx2_snd_pkt_handler(pfvf, cq,
-                                            &pfvf->qset.sq[cq->cint_idx],
-                                            cqe, budget, &tx_pkts, &tx_bytes);
+                       otx2_snd_pkt_handler(pfvf, cq, sq, cqe, budget,
+                                            &tx_pkts, &tx_bytes);
                }
+
                cqe->hdr.cqe_type = NIX_XQE_TYPE_INVALID;
                processed_cqe++;
                cq->pend_cqe--;
+
+               sq->cons_head++;
+               sq->cons_head &= (sq->sqe_cnt - 1);
        }

        /* Free CQEs to HW */
@@ -1072,17 +1078,17 @@ bool otx2_sq_append_skb(struct net_device *netdev, struct otx2_snd_queue *sq,
 {
        struct netdev_queue *txq = netdev_get_tx_queue(netdev, qidx);
        struct otx2_nic *pfvf = netdev_priv(netdev);
-       int offset, num_segs, free_sqe;
+       int offset, num_segs, free_desc;
        struct nix_sqe_hdr_s *sqe_hdr;

-       /* Check if there is room for new SQE.
-        * 'Num of SQBs freed to SQ's pool - SQ's Aura count'
-        * will give free SQE count.
+       /* Check if there is enough room between producer
+        * and consumer index.
         */
-       free_sqe = (sq->num_sqbs - *sq->aura_fc_addr) * sq->sqe_per_sqb;
+       free_desc = (sq->cons_head - sq->head - 1 + sq->sqe_cnt) & (sq->sqe_cnt - 1);
+       if (free_desc < sq->sqe_thresh)
+               return false;

-       if (free_sqe < sq->sqe_thresh ||
-           free_sqe < otx2_get_sqe_count(pfvf, skb))
+       if (free_desc < otx2_get_sqe_count(pfvf, skb))
                return false;

        num_segs = skb_shinfo(skb)->nr_frags + 1;
```

```
diff --git a/drivers/net/ethernet/marvell/octeontx2/nic/otx2_txrx.h b/drivers/net/ethernet/marvell/octeontx2/nic/otx2_txrx.h
index fbe62bbfb789af..93cac2c2664c20 100644
--- a/drivers/net/ethernet/marvell/octeontx2/nic/otx2_txrx.h
+++ b/drivers/net/ethernet/marvell/octeontx2/nic/otx2_txrx.h
@@ -79,6 +79,7 @@ struct sg_list {
 struct otx2_snd_queue {
        u8                      aura_id;
        u16                     head;
+       u16                     cons_head;
        u16                     sqe_size;
        u32                     sqe_cnt;
        u16                     num_sqbs;
```