author	Chen Jun <chenjun102@huawei.com>	2022-10-31 03:12:42 +0000
committer	Greg Kroah-Hartman <gregkh@linuxfoundation.org>	2022-11-10 18:17:29 +0100
commit	2dc97e15a54b7bdf457848aa8c663c98a24e58a6 (patch)	
tree	ea4a0ab3e8aeb9b4073fec2aabf431795b0f7e77	
parent	528677d3b4af985445bd4ac667485ded1ed11220 (diff)	
download	linux-2dc97e15a54b7bdf457848aa8c663c98a24e58a6.tar.gz	

blk-mq: Fix kmemleak in blk_mq_init_allocated_queue

```
[ Upstream commit 943f45b9399ed8b2b5190cbc797995edaa97f58f ]

There is a kmemleak caused by modprobe null_blk.ko

unreferenced object 0xffff8881acb1f000 (size 1024):
  comm "modprobe", pid 836, jiffies 4294971190 (age 27.068s)
  hex dump (first 32 bytes):
    00 00 00 00 ad 4e ad de ff ff ff ff 00 00 00 00  .....N..........
    ff ff ff ff ff ff ff ff 00 53 99 9e ff ff ff ff  .........S......
  backtrace:
    [<000000004a10c249>] kmalloc_node_trace+0x22/0x60
    [<00000000648f7950>] blk_mq_alloc_and_init_hctx+0x289/0x350
    [<00000000af06de0e>] blk_mq_realloc_hw_ctxs+0x2fe/0x3d0
    [<00000000e00c1872>] blk_mq_init_allocated_queue+0x48c/0x1440
    [<00000000d16b4e68>] __blk_mq_alloc_disk+0xc8/0x1c0
    [<00000000d10c98c3>] 0xffffffffc450d69d
    [<00000000b9299f48>] 0xffffffffc4538392
    [<0000000061c39ed6>] do_one_initcall+0xd0/0x4f0
    [<00000000b389383b>] do_init_module+0x1a4/0x680
    [<0000000087cf3542>] load_module+0x6249/0x7110
    [<00000000beba61b8>] __do_sys_finit_module+0x140/0x200
    [<00000000fdcfff51>] do_syscall_64+0x35/0x80
    [<000000003c0f1f71>] entry_SYSCALL_64_after_hwframe+0x46/0xb0

That is because q->ma_ops is set to NULL before blk_release_queue is
called.

blk_mq_init_queue_data
  blk_mq_init_allocated_queue
    blk_mq_realloc_hw_ctxs
      for (i = 0; i < set->nr_hw_queues; i++) {
        old_hctx = xa_load(&q->hctx_table, i);
        if (!blk_mq_alloc_and_init_hctx(.., i, ..))            [1]
          if (!old_hctx)
            break;

      xa_for_each_start(&q->hctx_table, j, hctx, j)
        blk_mq_exit_hctx(q, set, hctx, j);                     [2]

    if (!q->nr_hw_queues)                                      [3]
      goto err_hctxs;
```

```
  err_exit:
      q->mq_ops = NULL;                                           [4]

  blk_put_queue
    blk_release_queue
      if (queue_is_mq(q))                                         [5]
        blk_mq_release(q);

[1]: blk_mq_alloc_and_init_hctx failed at i != 0.
[2]: The hctxs allocated by [1] are moved to q->unused_hctx_list and
will be cleaned up in blk_mq_release.
[3]: q->nr_hw_queues is 0.
[4]: Set q->mq_ops to NULL.
[5]: queue_is_mq returns false due to [4]. And blk_mq_release
will not be called. The hctxs in q->unused_hctx_list are leaked.

To fix it, call blk_release_queue in exception path.

Fixes: 2f8f1336a48b ("blk-mq: always free hctx after request queue is freed")
Signed-off-by: Yuan Can <yuancan@huawei.com>
Signed-off-by: Chen Jun <chenjun102@huawei.com>
Reviewed-by: Ming Lei <ming.lei@redhat.com>
Link: https://lore.kernel.org/r/20221031031242.94107-1-chenjun102@huawei.com
Signed-off-by: Jens Axboe <axboe@kernel.dk>
Signed-off-by: Sasha Levin <sashal@kernel.org>
```

Diffstat

-rw-r--r-- block/blk-mq.c 4

1 files changed, 1 insertions, 3 deletions

```
diff --git a/block/blk-mq.c b/block/blk-mq.c
index fe840536e6ac4a..edf41959a705fc 100644
--- a/block/blk-mq.c
+++ b/block/blk-mq.c
@@ -4104,9 +4104,7 @@ int blk_mq_init_allocated_queue(struct blk_mq_tag_set *set,
        return 0;

 err_hctxs:
-       xa_destroy(&q->hctx_table);
-       q->nr_hw_queues = 0;
-       blk_mq_sysfs_deinit(q);
+       blk_mq_release(q);
 err_poll:
        blk_stat_free_callback(q->poll_cb);
        q->poll_cb = NULL;
```