



index : kernel/git/stable/linux.git

Linux kernel stable tree

master

switch

Stable Group

[about](#) [summary](#) [refs](#) [log](#) [tree](#) [commit](#) [diff](#) [stats](#)

log msg

search

author Jeff Hugo <quic_jhugo@quicinc.com> 2025-03-06 10:29:13 -0700
committer Greg Kroah-Hartman <gregkh@linuxfoundation.org> 2025-04-25 10:45:26 +0200
commit 5f084993c90d9d0b4a52a349ede5120f992a7ca1 (patch)
tree 66b9ad403518e2376c9f2710e7967db9c2285f9e
parent 7e439ff5efabb6db40a6fd37f17eef41199ddea9 (diff)
download linux-5f084993c90d9d0b4a52a349ede5120f992a7ca1.tar.gz

diff options

context: 3
space: include
mode: unified

bus: mhi: host: Fix race between unprepare and queue_buf

commit 0686a818d77a431fc3ba2fab4b46bbb04e8c9380 upstream.

A client driver may use mhi_unprepare_from_transfer() to quiesce incoming data during the client driver's tear down. The client driver might also be processing data at the same time, resulting in a call to mhi_queue_buf() which will invoke mhi_gen_tre(). If mhi_gen_tre() runs after mhi_unprepare_from_transfer() has torn down the channel, a panic will occur due to an invalid dereference leading to a page fault.

This occurs because mhi_gen_tre() does not verify the channel state after locking it. Fix this by having mhi_gen_tre() confirm the channel state is valid, or return error to avoid accessing deinitialized data.

Cc: stable@vger.kernel.org # 6.8

Fixes: b89b6a863dd5 ("bus: mhi: host: Add spinlock to protect WP access when queueing TREs")

Signed-off-by: Jeffrey Hugo <quic_jhugo@quicinc.com>

Signed-off-by: Jeff Hugo <jeff.hugo@oss.qualcomm.com>

Reviewed-by: Krishna Chaitanya Chundru <krishna.chundru@oss.qualcomm.com>

Reviewed-by: Youssef Samir <quic_yabdulra@quicinc.com>

Reviewed-by: Manivannan Sadhasivam <manivannan.sadhasivam@linaro.org>

Reviewed-by: Troy Hanson <quic_thanson@quicinc.com>

Link: <https://lore.kernel.org/r/20250306172913.856982-1-jeff.hugo@oss.qualcomm.com>

[mani: added stable tag]

Signed-off-by: Manivannan Sadhasivam <manivannan.sadhasivam@linaro.org>

Signed-off-by: Greg Kroah-Hartman <gregkh@linuxfoundation.org>

Diffstat

-rw-r--r-- drivers/bus/mhi/host/main.c 16

1 files changed, 10 insertions, 6 deletions

diff --git a/drivers/bus/mhi/host/main.c b/drivers/bus/mhi/host/main.c

index d6653cbcf94a2e..ad1e97222a0f49 100644

--- a/drivers/bus/mhi/host/main.c

+++ b/drivers/bus/mhi/host/main.c

```
@@ -1204,11 +1204,16 @@ int mhi_gen_tre(struct mhi_controller *mhi_cntrl, struct mhi_chan *mhi_chan,
    struct mhi_ring_element *mhi_tre;
    struct mhi_buf_info *buf_info;
    int eot, eob, chain, bei;
-   int ret;
+   int ret = 0;

    /* Protect accesses for reading and incrementing WP */
    write_lock_bh(&mhi_chan->lock);
```

```

+         if (mhi_chan->ch_state != MHI_CH_STATE_ENABLED) {
+             ret = -ENODEV;
+             goto out;
+         }
+
+         buf_ring = &mhi_chan->buf_ring;
+         tre_ring = &mhi_chan->tre_ring;
+
@@ -1226,10 +1231,8 @@ int mhi_gen_tre(struct mhi_controller *mhi_cntrl, struct mhi_chan *mhi_chan,
+
+         if (!info->pre_mapped) {
+             ret = mhi_cntrl->map_single(mhi_cntrl, buf_info);
-             if (ret) {
-                 write_unlock_bh(&mhi_chan->lock);
-                 return ret;
+             }
+             if (ret)
+                 goto out;
+         }
+
+         eob = !(flags & MHI_EOB);
@@ -1246,9 +1249,10 @@ int mhi_gen_tre(struct mhi_controller *mhi_cntrl, struct mhi_chan *mhi_chan,
+         mhi_add_ring_element(mhi_cntrl, tre_ring);
+         mhi_add_ring_element(mhi_cntrl, buf_ring);
+
+out:
+         write_unlock_bh(&mhi_chan->lock);
+
-         return 0;
+         return ret;
+     }
+
+     int mhi_queue_buf(struct mhi_device *mhi_dev, enum dma_data_direction dir,

```