author	Manjunatha Venkatesh <manjunatha.venkatesh@nxp.com>	2025-03-26 18:00:46 +0530
committer	Greg Kroah-Hartman <gregkh@linuxfoundation.org>	2025-04-20 10:15:44 +0200
commit	e6bba328578feb58c614c11868c259b40484c5fa (patch)	
tree	feca396c22b5cce05484c5359155b2655038a755	
parent	34aaf448e204b83bf59dbfb5f15d191d7ae51eee (diff)	
download	linux-e6bba328578feb58c614c11868c259b40484c5fa.tar.gz	

i3c: Add NULL pointer check in i3c_master_queue_ibi()

```
commit bd496a44f041da9ef3afe14d1d6193d460424e91 upstream.

The I3C master driver may receive an IBI from a target device that has not
been probed yet. In such cases, the master calls `i3c_master_queue_ibi()`
to queue an IBI work task, leading to "Unable to handle kernel read from
unreadable memory" and resulting in a kernel panic.

Typical IBI handling flow:
1. The I3C master scans target devices and probes their respective drivers.
2. The target device driver calls `i3c_device_request_ibi()` to enable IBI
   and assigns `dev->ibi = ibi`.
3. The I3C master receives an IBI from the target device and calls
   `i3c_master_queue_ibi()` to queue the target device driver's IBI
   handler task.

However, since target device events are asynchronous to the I3C probe
sequence, step 3 may occur before step 2, causing `dev->ibi` to be `NULL`,
leading to a kernel panic.

Add a NULL pointer check in `i3c_master_queue_ibi()` to prevent accessing
an uninitialized `dev->ibi`, ensuring stability.

Fixes: 3a379bbcea0af ("i3c: Add core I3C infrastructure")
Cc: stable@vger.kernel.org
Link: https://lore.kernel.org/lkml/Z9gjGYudiYyl3bSe@lizhi-Precision-Tower-5810/
Signed-off-by: Manjunatha Venkatesh <manjunatha.venkatesh@nxp.com>
Reviewed-by: Frank Li <Frank.Li@nxp.com>
Link: https://lore.kernel.org/r/20250326123047.2797946-1-manjunatha.venkatesh@nxp.com
Signed-off-by: Alexandre Belloni <alexandre.belloni@bootlin.com>
Signed-off-by: Greg Kroah-Hartman <gregkh@linuxfoundation.org>
```

Diffstat

-rw-r--r-- drivers/i3c/master.c 3

1 files changed, 3 insertions, 0 deletions

```
diff --git a/drivers/i3c/master.c b/drivers/i3c/master.c
index 53ab814b676ffd..7c1dc42b809bfc 100644
--- a/drivers/i3c/master.c
+++ b/drivers/i3c/master.c
@@ -2553,6 +2553,9 @@ static void i3c_master_unregister_i3c_devs(struct i3c_master_controller *master)
  */
 void i3c_master_queue_ibi(struct i3c_dev_desc *dev, struct i3c_ibi_slot *slot)
 {
+       if (!dev->ibi || !slot)
```

```
+                return;
+
         atomic_inc(&dev->ibi->pending_ibis);
         queue_work(dev->ibi->wq, &slot->work);
 }
```