# index : kernel/git/torvalds/linux.git

master switch

Linux kernel source tree

Linus Torvalds

about summary refs log tree commit diff stats

log msg search

path: root/kernel/dma/swiotlb.c

| | | |
|---|---|---|
| author | Halil Pasic <pasic@linux.ibm.com> | 2022-03-05 18:07:14 +0100 |
| committer | Linus Torvalds <torvalds@linux-foundation.org> | 2022-03-07 11:26:02 -0800 |
| commit | aa6f8dcbab473f3a3c7454b74caa46d36cdc5d13 (patch) | |
| tree | f4ff6004c13374d66d37db7221e120f8d409799c /kernel/dma/swiotlb.c | |
| parent | ffb217a13a2eaf6d5bd974fc83036a53ca69f1e2 (diff) | |
| download | linux-aa6f8dcbab473f3a3c7454b74caa46d36cdc5d13.tar.gz | |

**diff options**

| | |
|---|---|
| context: | 3 |
| space: | include |
| mode: | unified |

## swiotlb: rework "fix info leak with DMA_FROM_DEVICE"

```
Unfortunately, we ended up merging an old version of the patch "fix info
leak with DMA_FROM_DEVICE" instead of merging the latest one. Christoph
(the swiotlb maintainer), he asked me to create an incremental fix
(after I have pointed this out the mix up, and asked him for guidance).
So here we go.

The main differences between what we got and what was agreed are:
* swiotlb_sync_single_for_device is also required to do an extra bounce
* We decided not to introduce DMA_ATTR_OVERWRITE until we have exploiters
* The implantation of DMA_ATTR_OVERWRITE is flawed: DMA_ATTR_OVERWRITE
  must take precedence over DMA_ATTR_SKIP_CPU_SYNC

Thus this patch removes DMA_ATTR_OVERWRITE, and makes
swiotlb_sync_single_for_device() bounce unconditionally (that is, also
when dir == DMA_TO_DEVICE) in order do avoid synchronising back stale
data from the swiotlb buffer.

Let me note, that if the size used with dma_sync_* API is less than the
size used with dma_[un]map_*, under certain circumstances we may still
end up with swiotlb not being transparent. In that sense, this is no
perfect fix either.

To get this bullet proof, we would have to bounce the entire
mapping/bounce buffer. For that we would have to figure out the starting
address, and the size of the mapping in
swiotlb_sync_single_for_device(). While this does seem possible, there
seems to be no firm consensus on how things are supposed to work.

Signed-off-by: Halil Pasic <pasic@linux.ibm.com>
Fixes: ddbd89deb7d3 ("swiotlb: fix info leak with DMA_FROM_DEVICE")
Cc: stable@vger.kernel.org
Reviewed-by: Christoph Hellwig <hch@lst.de>
Signed-off-by: Linus Torvalds <torvalds@linux-foundation.org>
```

**Diffstat** **(limited to 'kernel/dma/swiotlb.c')**

| | | |
|---|---|---|
| -rw-r--r-- | kernel/dma/swiotlb.c | 23 |

1 files changed, 15 insertions, 8 deletions

```
diff --git a/kernel/dma/swiotlb.c b/kernel/dma/swiotlb.c
index bfc56cb217059..6db1c475ec827 100644
--- a/kernel/dma/swiotlb.c
+++ b/kernel/dma/swiotlb.c
@@ -627,10 +627,14 @@ phys_addr_t swiotlb_tbl_map_single(struct device *dev, phys_addr_t orig_addr,
 	for (i = 0; i < nr_slots(alloc_size + offset); i++)
 		mem->slots[index + i].orig_addr = slot_addr(orig_addr, i);
 	tlb_addr = slot_addr(mem->start, index) + offset;
-	if (!(attrs & DMA_ATTR_SKIP_CPU_SYNC) &&
-	    (!(attrs & DMA_ATTR_OVERWRITE) || dir == DMA_TO_DEVICE ||
-           dir == DMA_BIDIRECTIONAL))
-               swiotlb_bounce(dev, tlb_addr, mapping_size, DMA_TO_DEVICE);
+       /*
+        * When dir == DMA_FROM_DEVICE we could omit the copy from the orig
+        * to the tlb buffer, if we knew for sure the device will
+        * overwirte the entire current content. But we don't. Thus
+        * unconditional bounce may prevent leaking swiotlb content (i.e.
+        * kernel memory) to user-space.
+        */
+       swiotlb_bounce(dev, tlb_addr, mapping_size, DMA_TO_DEVICE);
        return tlb_addr;
 }

@@ -697,10 +701,13 @@ void swiotlb_tbl_unmap_single(struct device *dev, phys_addr_t tlb_addr,
 void swiotlb_sync_single_for_device(struct device *dev, phys_addr_t tlb_addr,
                size_t size, enum dma_data_direction dir)
 {
-       if (dir == DMA_TO_DEVICE || dir == DMA_BIDIRECTIONAL)
-               swiotlb_bounce(dev, tlb_addr, size, DMA_TO_DEVICE);
-       else
-               BUG_ON(dir != DMA_FROM_DEVICE);
+       /*
+        * Unconditional bounce is necessary to avoid corruption on
+        * sync_*_for_cpu or dma_ummap_* when the device didn't overwrite
+        * the whole lengt of the bounce buffer.
+        */
+       swiotlb_bounce(dev, tlb_addr, size, DMA_TO_DEVICE);
+       BUG_ON(!valid_dma_direction(dir));
 }

 void swiotlb_sync_single_for_cpu(struct device *dev, phys_addr_t tlb_addr,
```

generated by cgit (git 2.34.1) at 2022-11-26 23:16:52 +0000