author	Chris Bainbridge <chris.bainbridge@gmail.com>	2025-03-26 12:52:10 +0000
committer	Greg Kroah-Hartman <gregkh@linuxfoundation.org>	2025-04-25 10:43:58 +0200
commit	12b038d521c75e3521522503becf3bc162628469 (patch)	
tree	99d8a04e63d41ce66896cfa97cc24f3580b690c4	
parent	6785702f4a6dcd1440518ca8650683ecd0c01b8d (diff)	
download	linux-12b038d521c75e3521522503becf3bc162628469.tar.gz	

diff options
context: 3
space: include
mode: unified

drm/nouveau: prime: fix ttm_bo_delayed_delete oops

```
commit 8ec0fbb28d049273bfd4f1e7a5ae4c74884beed3 upstream.

Fix an oops in ttm_bo_delayed_delete which results from dererencing a
dangling pointer:

Oops: general protection fault, probably for non-canonical address 0x6b6b6b6b6b6b6b7b: 0000 [#1] PREEMPT SMP
CPU: 4 UID: 0 PID: 1082 Comm: kworker/u65:2 Not tainted 6.14.0-rc4-00267-g505460b44513-dirty #216
Hardware name: LENOVO 82N6/LNVNB161216, BIOS GKCN65WW 01/16/2024
Workqueue: ttm ttm_bo_delayed_delete [ttm]
RIP: 0010:dma_resv_iter_first_unlocked+0x55/0x290
Code: 31 f6 48 c7 c7 00 2b fa aa e8 97 bd 52 ff e8 a2 c1 53 00 5a 85 c0 74 48 e9 88 01 00 00 4c 89 63 20 4d 85 e4 0f 84 30 01 00 00 <41> 8b 44 24 10
RSP: 0018:ffffbf9383473d60 EFLAGS: 00010202
RAX: 0000000000000001 RBX: ffffbf9383473d88 RCX: 0000000000000000
RDX: 0000000000000000 RSI: 0000000000000000 RDI: 0000000000000000
RBP: ffffbf9383473d78 R08: 0000000000000000 R09: 0000000000000000
R10: 0000000000000000 R11: 0000000000000000 R12: 6b6b6b6b6b6b6b6b
R13: ffffa003bbf78580 R14: ffffa003a6728040 R15: 00000000000383cc
FS:  0000000000000000(0000) GS:ffffa00991c00000(0000) knlGS:0000000000000000
CS:  0010 DS: 0000 ES: 0000 CR0: 0000000080050033
CR2: 0000758348024dd0 CR3: 000000012c259000 CR4: 0000000000f50ef0
PKRU: 55555554
Call Trace:
 <TASK>
 ? __die_body.cold+0x19/0x26
 ? die_addr+0x3d/0x70
 ? exc_general_protection+0x159/0x460
 ? asm_exc_general_protection+0x27/0x30
 ? dma_resv_iter_first_unlocked+0x55/0x290
 dma_resv_wait_timeout+0x56/0x100
 ttm_bo_delayed_delete+0x69/0xb0 [ttm]
 process_one_work+0x217/0x5c0
 worker_thread+0x1c8/0x3d0
 ? apply_wqattrs_cleanup.part.0+0xc0/0xc0
 kthread+0x10b/0x240
 ? kthreads_online_cpu+0x140/0x140
 ret_from_fork+0x40/0x70
 ? kthreads_online_cpu+0x140/0x140
 ret_from_fork_asm+0x11/0x20
 </TASK>

The cause of this is:

- drm_prime_gem_destroy calls dma_buf_put(dma_buf) which releases the
  reference to the shared dma_buf. The reference count is 0, so the
  dma_buf is destroyed, which in turn decrements the corresponding
  amdgpu_bo reference count to 0, and the amdgpu_bo is destroyed -
  calling drm_gem_object_release then dma_resv_fini (which destroys the
  reservation object), then finally freeing the amdgpu_bo.

- nouveau_bo obj->bo.base.resv is now a dangling pointer to the memory
  formerly allocated to the amdgpu_bo.

- nouveau_gem_object_del calls ttm_bo_put(&nvbo->bo) which calls
  ttm_bo_release, which schedules ttm_bo_delayed_delete.

- ttm_bo_delayed_delete runs and dereferences the dangling resv pointer,
  resulting in a general protection fault.

Fix this by moving the drm_prime_gem_destroy call from
nouveau_gem_object_del to nouveau_bo_del_ttm. This ensures that it will
be run after ttm_bo_delayed_delete.

Signed-off-by: Chris Bainbridge <chris.bainbridge@gmail.com>
Suggested-by: Christian König <christian.koenig@amd.com>
Fixes: 22b33e8ed0e3 ("nouveau: add PRIME support")
Closes: https://gitlab.freedesktop.org/drm/amd/-/issues/3937
Cc: Stable@vger.kernel.org
Signed-off-by: Danilo Krummrich <dakr@kernel.org>
Link: https://patchwork.freedesktop.org/patch/msgid/Z-P4epVK8k7tFZ7C@debian.local
Signed-off-by: Greg Kroah-Hartman <gregkh@linuxfoundation.org>
```

Diffstat

-rw-r--r-- drivers/gpu/drm/nouveau/nouveau_bo.c 3
-rw-r--r-- drivers/gpu/drm/nouveau/nouveau_gem.c 3

2 files changed, 3 insertions, 3 deletions

```
diff --git a/drivers/gpu/drm/nouveau/nouveau_bo.c b/drivers/gpu/drm/nouveau/nouveau_bo.c
index f2dca41e46c5fb..2953d6a6d39080 100644
--- a/drivers/gpu/drm/nouveau/nouveau_bo.c
+++ b/drivers/gpu/drm/nouveau/nouveau_bo.c
@@ -143,6 +143,9 @@ nouveau_bo_del_ttm(struct ttm_buffer_object *bo)
        nouveau_bo_del_io_reserve_lru(bo);
        nv10_bo_put_tile_region(dev, nvbo->tile, NULL);

+       if (bo->base.import_attach)
+               drm_prime_gem_destroy(&bo->base, bo->sg);
+
        /*
         * If nouveau_bo_new() allocated this buffer, the GEM object was never
         * initialized, so don't attempt to release it.
```

```
diff --git a/drivers/gpu/drm/nouveau/nouveau_gem.c b/drivers/gpu/drm/nouveau/nouveau_gem.c
index fab542a758ff97..a14728427ee434 100644
--- a/drivers/gpu/drm/nouveau/nouveau_gem.c
+++ b/drivers/gpu/drm/nouveau/nouveau_gem.c
@@ -87,9 +87,6 @@ nouveau_gem_object_del(struct drm_gem_object *gem)
                return;
        }

-       if (gem->import_attach)
-               drm_prime_gem_destroy(gem, nvbo->bo.sg);
-
        ttm_bo_put(&nvbo->bo);

        pm_runtime_mark_last_busy(dev);
```