From: Jeremy Cline <jcline@redhat.com>
To: Ben Skeggs <bskeggs@redhat.com>
Cc: Karol Herbst <kherbst@redhat.com>,
David Airlie <airlied@linux.ie>,
nouveau@lists.freedesktop.org, linux-kernel@vger.kernel.org,
dri-devel@lists.freedesktop.org, Jeremy Cline <jcline@redhat.com>
Subject: [PATCH 3/3] drm/nouveau: clean up all clients on device removal
Date: Tue, 3 Nov 2020 14:49:12 -0500 [thread overview]
Message-ID: <20201103194912.184413-4-jcline@redhat.com> (raw)
In-Reply-To: <20201103194912.184413-1-jcline@redhat.com>

The postclose handler can run after the device has been removed (or the
driver has been unbound) since userspace clients are free to hold the
file open the file as long as they want. Because the device removal
callback frees the entire nouveau_drm structure, any reference to it in
the postclose handler will result in a use-after-free.

To reproduce this, one must simply open the device file, unbind the
driver (or physically remove the device), and then close the device
file. This was found and can be reproduced easily with the IGT
core_hotunplug tests.

To avoid this, all clients are cleaned up in the device finialization
rather than deferring it to the postclose handler, and the postclose
handler is protected by a critical section which ensures the
drm_dev_unplug() and the postclose handler won't race.

This is not an ideal fix, since as I understand the proposed plan for
the kernel<->userspace interface for hotplug support, destroying the
client before the file is closed will cause problems. However, I believe
to properly fix this issue, the lifetime of the nouveau_drm structure
needs to be extended to match the drm_device, and this proved to be a
rather invasive change. Thus, I've broken this out so the fix can be
easily backported.

Signed-off-by: Jeremy Cline <jcline@redhat.com>
---
drivers/gpu/drm/nouveau/nouveau_drm.c | 30 ++++++++++++++++++++++++++++
1 file changed, 30 insertions(+)

```
diff --git a/drivers/gpu/drm/nouveau/nouveau_drm.c b/drivers/gpu/drm/nouveau/nouveau_drm.c
index d182b877258a..74fab777f4d0 100644
--- a/drivers/gpu/drm/nouveau/nouveau_drm.c
+++ b/drivers/gpu/drm/nouveau/nouveau_drm.c
@@ -628,6 +628,7 @@ nouveau_drm_device_init(struct drm_device *dev)
 static void
 nouveau_drm_device_fini(struct drm_device *dev)
 {
+       struct nouveau_cli *cli, *temp_cli;
        struct nouveau_drm *drm = nouveau_drm(dev);

        if (nouveau_pmops_runtime()) {
@@ -652,6 +653,24 @@ nouveau_drm_device_fini(struct drm_device *dev)
        nouveau_ttm_fini(drm);
        nouveau_vga_fini(drm);

+       /*
+        * There may be existing clients from as-yet unclosed files. For now,
+        * clean them up here rather than deferring until the file is closed,
+        * but this likely not correct if we want to support hot-unplugging
+        * properly.
+        */
+       mutex_lock(&drm->clients_lock);
+       list_for_each_entry_safe(cli, temp_cli, &drm->clients, head) {
+               list_del(&cli->head);
+               mutex_lock(&cli->mutex);
+               if (cli->abi16)
+                       nouveau_abi16_fini(cli->abi16);
+               mutex_unlock(&cli->mutex);
+               nouveau_cli_fini(cli);
+               kfree(cli);
+       }
+       mutex_unlock(&drm->clients_lock);
+
        nouveau_cli_fini(&drm->client);
        nouveau_cli_fini(&drm->master);
        nvif_parent_dtor(&drm->parent);
@@ -1110,6 +1129,16 @@ nouveau_drm_postclose(struct drm_device *dev, struct drm_file *fpriv)
 {
        struct nouveau_cli *cli = nouveau_cli(fpriv);
        struct nouveau_drm *drm = nouveau_drm(dev);
+       int dev_index;
+
+       /*
+        * The device is gone, and as it currently stands all clients are
+        * cleaned up in the removal codepath. In the future this may change
+        * so that we can support hot-unplugging, but for now we immediately
+        * return to avoid a double-free situation.
+        */
+       if (!drm_dev_enter(dev, &dev_index))
+               return;

        pm_runtime_get_sync(dev->dev);

@@ -1126,6 +1155,7 @@ nouveau_drm_postclose(struct drm_device *dev, struct drm_file *fpriv)
        kfree(cli);
        pm_runtime_mark_last_busy(dev->dev);
        pm_runtime_put_autosuspend(dev->dev);
+       drm_dev_exit(dev_index);
 }

 static const struct drm_ioctl_desc
--
2.28.0
```


---

next prev parent reply other threads:[~2020-11-03 19:49 UTC|newest]

Thread overview: 16+ messages / expand[flat|nested] mbox.gz Atom feed top
2020-11-03 19:49 [PATCH 0/3] drm/nouveau: fix a use-after-free in postclose() Jeremy Cline
2020-11-03 19:49 ` [PATCH 1/3] drm/nouveau: use drm_dev_unplug() during device removal Jeremy Cline
2020-11-03 19:49 ` [PATCH 2/3] drm/nouveau: Add a dedicated mutex for the clients list Jeremy Cline
2020-11-25 18:37 ` Lyude Paul
2020-11-25 19:45 ` Jeremy Cline
2020-11-03 19:49 ` Jeremy Cline [this message]
2020-11-25 18:44 ` [PATCH 3/3] drm/nouveau: clean up all clients on device removal Lyude Paul
2020-11-25 20:26 ` [PATCH v2 0/3] drm/nouveau: fix a use-after-free in postclose() Jeremy Cline
2020-11-25 20:26 ` [PATCH v2 1/3] drm/nouveau: use drm_dev_unplug() during device removal Jeremy Cline
2020-11-25 20:26 ` [PATCH v2 2/3] drm/nouveau: Add a dedicated mutex for the clients list Jeremy Cline
2020-11-25 20:26 ` [PATCH v2 3/3] drm/nouveau: clean up all clients on device removal Jeremy Cline
2021-03-26 22:00 ` [PATCH v2 0/3] drm/nouveau: fix a use-after-free in postclose() Lyude Paul
2021-08-16 7:03 ` Salvatore Bonaccorso
2021-08-17 20:32 ` Lyude Paul
2021-10-11 7:05 ` Salvatore Bonaccorso
2021-10-11 11:05 ` Karol Herbst

find likely ancestor, descendant, or conflicting patches for this message:
dfblob:d182b877258 dfblob:74fab777f4d
search (help)

Reply instructions:

You may reply publicly to this message via plain-text email
using any one of the following methods:

* Save the following mbox file, import it into your mail client,
  and reply-to-all from there: mbox

  Avoid top-posting and favor interleaved quoting:
  https://en.wikipedia.org/wiki/Posting_style#Interleaved_style

* Reply using the **--to, --cc**, and **--in-reply-to**
  switches of git-send-email(1):

  git send-email \
    --in-reply-to=20201103194912.184413-4-jcline@redhat.com \
    --to=jcline@redhat.com \
    --cc=airlied@linux.ie \
    --cc=bskeggs@redhat.com \
    --cc=dri-devel@lists.freedesktop.org \
    --cc=kherbst@redhat.com \
    --cc=linux-kernel@vger.kernel.org \
    --cc=nouveau@lists.freedesktop.org \
    /path/to/YOUR_REPLY

  https://kernel.org/pub/software/scm/git/docs/git-send-email.html

* If your mail client supports setting the **In-Reply-To** header
  via mailto: links, try the mailto: link

Be sure your reply has a **Subject:** header at the top and a blank line before the message body.
