From: Haimin Zhang <tcs.kernel@gmail.com>
To: Jens Axboe <axboe@kernel.dk>, linux-block@vger.kernel.org
Cc: Haimin Zhang <tcs.kernel@gmail.com>
Subject: [PATCH] block-map: add __GFP_ZERO flag for alloc_page in function bio_copy_kern
Date: Wed, 16 Feb 2022 16:40:38 +0800 [thread overview]
Message-ID: <20220216084038.15635-1-tcs.kernel@gmail.com> (raw)

Add __GFP_ZERO flag for alloc_page in function bio_copy_kern to initialize
the buffer of a bio.

Signed-off-by: Haimin Zhang <tcs.kernel@gmail.com>

This can cause a kernel-info-leak problem.
0. This problem occurred in function scsi_ioctl. If the parameter cmd is
SCSI_IOCTL_SEND_COMMAND, the function scsi_ioctl will call sg_scsi_ioctl to further process.
1. In function sg_scsi_ioctl, it creates a scsi request and calls blk_rq_map_kern to map
kernel data to a request.
3. blq_rq_map_kern calls bio_copy_kern to request a bio.
4. bio_copy_kern calls alloc_page to request the buffer of a bio. In the case of reading, it
wouldn't fill anything into the buffer.

```
__alloc_pages+0xbbf/0x1090 build/../mm/page_alloc.c:5409
alloc_pages+0x8a5/0xb80
bio_copy_kern build/../block/blk-map.c:449 [inline]
blk_rq_map_kern+0x813/0x1400 build/../block/blk-map.c:640
sg_scsi_ioctl build/../drivers/scsi/scsi_ioctl.c:618 [inline]
scsi_ioctl+0x40c0/0x4600 build/../drivers/scsi/scsi_ioctl.c:932
sg_ioctl_common build/../drivers/scsi/sg.c:1112 [inline]
sg_ioctl+0x3351/0x4c10 build/../drivers/scsi/sg.c:1165
vfs_ioctl build/../fs/ioctl.c:51 [inline]
__do_sys_ioctl build/../fs/ioctl.c:874 [inline]
__se_sys_ioctl+0x2df/0x4a0 build/../fs/ioctl.c:860
__x64_sys_ioctl+0xd8/0x110 build/../fs/ioctl.c:860
do_syscall_x64 build/../arch/x86/entry/common.c:51 [inline]
do_syscall_64+0x54/0xd0 build/../arch/x86/entry/common.c:82
entry_SYSCALL_64_after_hwframe+0x44/0xae
```

5. Then this request will be sent to the disk driver. When bio is finished,
bio_copy_kern_endio_read will copy the readed content back to parameter data from the bio.
But if the block driver didn't process this request, the buffer of bio is still unitialized.

```
memcpy_from_page build/../include/linux/highmem.h:346 [inline]
memcpy_from_bvec build/../include/linux/bvec.h:207 [inline]
bio_copy_kern_endio_read+0x4a3/0x620 build/../block/blk-map.c:403
bio_endio+0xa7f/0xac0 build/../block/bio.c:1491
req_bio_endio build/../block/blk-mq.c:674 [inline]
blk_update_request+0x1129/0x22d0 build/../block/blk-mq.c:742
scsi_end_request+0x119/0xe40 build/../drivers/scsi/scsi_lib.c:543
scsi_io_completion+0x329/0x810 build/../drivers/scsi/scsi_lib.c:939
scsi_finish_command+0x6e3/0x700 build/../drivers/scsi/scsi.c:199
scsi_complete+0x239/0x640 build/../drivers/scsi/scsi_lib.c:1441
blk_complete_reqs build/../block/blk-mq.c:892 [inline]
blk_done_softirq+0x189/0x260 build/../block/blk-mq.c:897
__do_softirq+0x1ee/0x7c5 build/../kernel/softirq.c:558
```

6. Finally, the internal buffer's content is copied to the user buffer which is specified by
the parameter sic->data of sg_scsi_ioctl.

```
_copy_to_user+0x1c9/0x270 build/../lib/usercopy.c:33
copy_to_user build/../include/linux/uaccess.h:209 [inline]
sg_scsi_ioctl build/../drivers/scsi/scsi_ioctl.c:634 [inline]
scsi_ioctl+0x44d9/0x4600 build/../drivers/scsi/scsi_ioctl.c:932
sg_ioctl_common build/../drivers/scsi/sg.c:1112 [inline]
sg_ioctl+0x3351/0x4c10 build/../drivers/scsi/sg.c:1165
vfs_ioctl build/../fs/ioctl.c:51 [inline]
__do_sys_ioctl build/../fs/ioctl.c:874 [inline]
__se_sys_ioctl+0x2df/0x4a0 build/../fs/ioctl.c:860
__x64_sys_ioctl+0xd8/0x110 build/../fs/ioctl.c:860
do_syscall_x64 build/../arch/x86/entry/common.c:51 [inline]
do_syscall_64+0x54/0xd0 build/../arch/x86/entry/common.c:82
entry_SYSCALL_64_after_hwframe+0x44/0xae
 block/blk-map.c | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)

diff --git a/block/blk-map.c b/block/blk-map.c
index 4526adde0156..c7f71d83eff1 100644
--- a/block/blk-map.c
+++ b/block/blk-map.c
@@ -446,7 +446,7 @@ static struct bio *bio_copy_kern(struct request_queue *q, void *data,
                if (bytes > len)
                        bytes = len;

-               page = alloc_page(GFP_NOIO | gfp_mask);
+               page = alloc_page(GFP_NOIO | __GFP_ZERO | gfp_mask);
                if (!page)
                        goto cleanup;

--
2.30.1 (Apple Git-130)
```

---

switches of git-send-email(1):

```
git send-email \
  --in-reply-to=20220216084038.15635-1-tcs.kernel@gmail.com \
  --to=tcs.kernel@gmail.com \
  --cc=axboe@kernel.dk \
  --cc=linux-block@vger.kernel.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.
