From Jason Yan <>
Subject [PATCH v4] block: Fix use-after-free in blkdev_get()
Date Mon, 8 Jun 2020 10:05:57 +0800

In blkdev_get() we call __blkdev_get() to do some internal jobs and if there is some errors in __blkdev_get(), the bdput() is called which means we have released the refcount of the bdev (actually the refcount of the bdev inode). This means we cannot access bdev after that point. But acctually bdev is still accessed in blkdev_get() after calling __blkdev_get(). This results in use-after-free if the refcount is the last one we released in __blkdev_get(). Let's take a look at the following scenerio:

```
  CPU0            CPU1                    CPU2
blkdev_open     blkdev_open           Remove disk
                  bd_acquire
                  blkdev_get
                    __blkdev_get      del_gendisk
                                        bdev_unhash_inode
  bd_acquire          bdev_get_gendisk
    bd_forget           failed because of unhashed
          bdput
                      bdput (the last one)
                        bdev_evict_inode

                    access bdev => use after free
```

```
[  459.350216] BUG: KASAN: use-after-free in __lock_acquire+0x24c1/0x31b0
[  459.351190] Read of size 8 at addr ffff88806c815a80 by task syz-executor.0/20132
[  459.352347]
[  459.352594] CPU: 0 PID: 20132 Comm: syz-executor.0 Not tainted 4.19.90 #2
[  459.353628] Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS 1.10.2-1ubuntu1 04/01/2014
[  459.354947] Call Trace:
[  459.355337]  dump_stack+0x111/0x19e
[  459.355879]  ? __lock_acquire+0x24c1/0x31b0
[  459.356523]  print_address_description+0x60/0x223
[  459.357248]  ? __lock_acquire+0x24c1/0x31b0
[  459.357887]  kasan_report.cold+0xae/0x2d8
[  459.358503]  __lock_acquire+0x24c1/0x31b0
[  459.359120]  ? _raw_spin_unlock_irq+0x24/0x40
[  459.359784]  ? lockdep_hardirqs_on+0x37b/0x580
[  459.360465]  ? _raw_spin_unlock_irq+0x24/0x40
[  459.361123]  ? finish_task_switch+0x125/0x600
[  459.361812]  ? finish_task_switch+0xee/0x600
[  459.362471]  ? mark_held_locks+0xf0/0xf0
[  459.363108]  ? __schedule+0x96f/0x21d0
[  459.363716]  lock_acquire+0x111/0x320
[  459.364285]  ? blkdev_get+0xce/0xbe0
[  459.364846]  ? blkdev_get+0xce/0xbe0
[  459.365390]  __mutex_lock+0xf9/0x12a0
[  459.365948]  ? blkdev_get+0xce/0xbe0
[  459.366493]  ? bdev_evict_inode+0x1f0/0x1f0
[  459.367130]  ? blkdev_get+0xce/0xbe0
[  459.367678]  ? destroy_inode+0xbc/0x110
[  459.368261]  ? mutex_trylock+0x1a0/0x1a0
[  459.368867]  ? __blkdev_get+0x3e6/0x1280
[  459.369463]  ? bdev_disk_changed+0x1d0/0x1d0
[  459.370114]  ? blkdev_get+0xce/0xbe0
[  459.370656]  blkdev_get+0xce/0xbe0
[  459.371178]  ? find_held_lock+0x2c/0x110
[  459.371774]  ? __blkdev_get+0x1280/0x1280
[  459.372383]  ? lock_downgrade+0x680/0x680
[  459.373002]  ? lock_acquire+0x111/0x320
[  459.373587]  ? bd_acquire+0x21/0x2c0
[  459.374134]  ? do_raw_spin_unlock+0x4f/0x250
[  459.374780]  blkdev_open+0x202/0x290
[  459.375325]  do_dentry_open+0x49e/0x1050
[  459.375924]  ? blkdev_get_by_dev+0x70/0x70
[  459.376543]  ? __x64_sys_fchdir+0x1f0/0x1f0
[  459.377192]  ? inode_permission+0xbe/0x3a0
[  459.377818]  path_openat+0x148c/0x3f50
[  459.378392]  ? kmem_cache_alloc+0xd5/0x280
[  459.379016]  ? entry_SYSCALL_64_after_hwframe+0x49/0xbe
[  459.379802]  ? path_lookupat.isra.0+0x900/0x900
[  459.380489]  ? __lock_is_held+0xad/0x140
[  459.381093]  do_filp_open+0x1a1/0x280
[  459.381654]  ? may_open_dev+0xf0/0xf0
[  459.382214]  ? find_held_lock+0x2c/0x110
[  459.382816]  ? lock_downgrade+0x680/0x680
[  459.383425]  ? __lock_is_held+0xad/0x140
[  459.384024]  ? do_raw_spin_unlock+0x4f/0x250
[  459.384668]  ? _raw_spin_unlock+0x1f/0x30
[  459.385280]  ? __alloc_fd+0x448/0x560
[  459.385841]  do_sys_open+0x3c3/0x500
[  459.386386]  ? filp_open+0x70/0x70
[  459.386911]  ? trace_hardirqs_on_thunk+0x1a/0x1c
[  459.387610]  ? trace_hardirqs_off_caller+0x55/0x1c0
[  459.388342]  ? do_syscall_64+0x1a/0x520
[  459.388930]  do_syscall_64+0xc3/0x520
[  459.389490]  entry_SYSCALL_64_after_hwframe+0x49/0xbe
[  459.390248] RIP: 0033:0x416211
[  459.390720] Code: 75 14 b8 02 00 00 00 0f 05 48 3d 01 f0 ff ff 0f 83
04 19 00 00 c3 48 83 ec 08 e8 0a fa ff ff 48 89 04 24 b8 02 00 00 00 0f
   05 <48> 8b 3c 24 48 89 c2 e8 53 fa ff ff 48 89 d0 48 83 c4 08 48 3d
      01
[  459.393483] RSP: 002b:00007fe45dfe9a60 EFLAGS: 00000293 ORIG_RAX: 0000000000000002
[  459.394610] RAX: ffffffffffffffda RBX: 00007fe45dfea6d4 RCX: 0000000000416211
[  459.395678] RDX: 00007fe45dfe9b0a RSI: 0000000000000002 RDI: 00007fe45dfe9b00
[  459.396758] RBP: 000000000076bf20 R08: 0000000000000000 R09: 000000000000000a
[  459.397930] R10: 0000000000000075 R11: 0000000000000293 R12: 00000000ffffffff
[  459.399022] R13: 00000000000000bd9 R14: 00000000004cdb80 R15: 000000000076bf2c
[  459.400168]
[  459.400430] Allocated by task 20132:
[  459.401038]  kasan_kmalloc+0xbf/0xe0
[  459.401652]  kmem_cache_alloc+0xd5/0x280
[  459.402330]  bdev_alloc_inode+0x18/0x40
[  459.402970]  alloc_inode+0x5f/0x180
[  459.403510]  iget5_locked+0x57/0xd0
[  459.404095]  bdget+0x94/0x4e0
[  459.404607]  bd_acquire+0xfa/0x2c0
[  459.405113]  blkdev_open+0x110/0x290
[  459.405702]  do_dentry_open+0x49e/0x1050
[  459.406340]  path_openat+0x148c/0x3f50
[  459.406926]  do_filp_open+0x1a1/0x280
[  459.407471]  do_sys_open+0x3c3/0x500
[  459.408010]  do_syscall_64+0xc3/0x520
[  459.408572]  entry_SYSCALL_64_after_hwframe+0x49/0xbe
[  459.409415]
[  459.409679] Freed by task 1262:
[  459.410212]  __kasan_slab_free+0x129/0x170
[  459.410919]  kmem_cache_free+0xb2/0x2a0
[  459.411564]  rcu_process_callbacks+0xbb2/0x2320
[  459.412318]  __do_softirq+0x225/0x8ac
```

Fix this by delaying bdput() to the end of blkdev_get() which means we have finished accessing bdev.

Cc: Christoph Hellwig <hch@lst.de>
Cc: Jens Axboe <axboe@kernel.dk>
Cc: Ming Lei <ming.lei@redhat.com>

```
Cc: Jan Kara <jack@suse.cz>
Reported-by: Hulk Robot <hulkci@huawei.com>
Signed-off-by: Jason Yan <yanaijie@huawei.com>
Reviewed-by: Jan Kara <jack@suse.cz>
---
 v4: Remove uneeded braces and add Reviewed-by tag from Jan Kara.
 v3: Add bdput() when __blkdev_get() calling itself failed.
 v2: Add Reported-by tag and cc linux-block mailing list

 fs/block_dev.c | 12 ++++++++-----
 1 file changed, 7 insertions(+), 5 deletions(-)

diff --git a/fs/block_dev.c b/fs/block_dev.c
index 47860e589388..08c87db3a92b 100644
--- a/fs/block_dev.c
+++ b/fs/block_dev.c
@@ -1565,10 +1565,8 @@ static int __blkdev_get(struct block_device *bdev, fmode_t mode, int for_part)
 	 */
 	if (!for_part) {
 		ret = devcgroup_inode_permission(bdev->bd_inode, perm);
-		if (ret != 0) {
-			bdput(bdev);
+		if (ret != 0)
 			return ret;
-		}
 	}

  restart:
@@ -1637,8 +1635,10 @@ static int __blkdev_get(struct block_device *bdev, fmode_t mode, int for_part)
 			BUG_ON(for_part);
 			ret = __blkdev_get(whole, mode, 1);
-			if (ret)
+			if (ret) {
+				bdput(whole);
 				goto out_clear;
+			}
 			bdev->bd_contains = whole;
 			bdev->bd_part = disk_get_part(disk, partno);
 			if (!(disk->flags & GENHD_FL_UP) ||
@@ -1688,7 +1688,6 @@ static int __blkdev_get(struct block_device *bdev, fmode_t mode, int for_part)
 	disk_unblock_events(disk);
 	put_disk_and_module(disk);
  out:
-	bdput(bdev);

 	return ret;
 }
@@ -1755,6 +1754,9 @@ int blkdev_get(struct block_device *bdev, fmode_t mode, void *holder)
 		bdput(whole);
 	}

+	if (res)
+		bdput(bdev);
+
 	return res;
 }
 EXPORT_SYMBOL(blkdev_get);
--
2.21.3
```