author	Ryusuke Konishi <konishi.ryusuke@gmail.com>	2022-11-04 23:29:59 +0900
committer	Greg Kroah-Hartman <gregkh@linuxfoundation.org>	2022-11-25 17:42:09 +0100
commit	9b162e81045266a2d5b44df9dffdf05c54de9cca (patch)	
tree	83df0511641d69bc3a1adcbf5d6f9ff7fa8633fe	
parent	36ff974b0310771417c0be64b64aa221bd70d63d (diff)	
download	linux-9b162e81045266a2d5b44df9dffdf05c54de9cca.tar.gz	

diff options

context: 3

space: include

mode: unified

nilfs2: fix use-after-free bug of ns_writer on remount

commit 8cccf05fe857a18ee26e20d11a8455a73ffd4efd upstream.

If a nilfs2 filesystem is downgraded to read-only due to metadata corruption on disk and is remounted read/write, or if emergency read-only remount is performed, detaching a log writer and synchronizing the filesystem can be done at the same time.

In these cases, use-after-free of the log writer (hereinafter nilfs->ns_writer) can happen as shown in the scenario below:

```
 Task1                                Task2
 ---------------------------------    -----------------------------
 nilfs_construct_segment
   nilfs_segctor_sync
     init_wait
     init_waitqueue_entry
     add_wait_queue
     schedule
                                      nilfs_remount (R/W remount case)
                                        nilfs_attach_log_writer
                                          nilfs_detach_log_writer
                                            nilfs_segctor_destroy
                                              kfree
     finish_wait
       _raw_spin_lock_irqsave
         __raw_spin_lock_irqsave
           do_raw_spin_lock
             debug_spin_lock_before  <-- use-after-free
```

While Task1 is sleeping, nilfs->ns_writer is freed by Task2. After Task1 waked up, Task1 accesses nilfs->ns_writer which is already freed. This scenario diagram is based on the Shigeru Yoshida's post [1].

This patch fixes the issue by not detaching nilfs->ns_writer on remount so that this UAF race doesn't happen. Along with this change, this patch also inserts a few necessary read-only checks with superblock instance where only the ns_writer pointer was used to check if the filesystem is read-only.

Link: https://syzkaller.appspot.com/bug?id=79a4c002e960419ca173d55e863bd09e8112df8b
Link: https://lkml.kernel.org/r/20221103141759.1836312-1-syoshida@redhat.com [1]
Link: https://lkml.kernel.org/r/20221104142959.28296-1-konishi.ryusuke@gmail.com
Signed-off-by: Ryusuke Konishi <konishi.ryusuke@gmail.com>

```
Reported-by: syzbot+f816fa82f8783f7a02bb@syzkaller.appspotmail.com
Reported-by: Shigeru Yoshida <syoshida@redhat.com>
Tested-by: Ryusuke Konishi <konishi.ryusuke@gmail.com>
Cc: <stable@vger.kernel.org>
Signed-off-by: Andrew Morton <akpm@linux-foundation.org>
Signed-off-by: Greg Kroah-Hartman <gregkh@linuxfoundation.org>
```

Diffstat

```
-rw-r--r-- fs/nilfs2/segment.c 15
-rw-r--r-- fs/nilfs2/super.c    2
```

2 files changed, 8 insertions, 9 deletions

```
diff --git a/fs/nilfs2/segment.c b/fs/nilfs2/segment.c
index 535543ab4e26aa..11914b3585b34f 100644
--- a/fs/nilfs2/segment.c
+++ b/fs/nilfs2/segment.c
@@ -322,7 +322,7 @@ void nilfs_relax_pressure_in_lock(struct super_block *sb)
         struct the_nilfs *nilfs = sb->s_fs_info;
         struct nilfs_sc_info *sci = nilfs->ns_writer;

-        if (!sci || !sci->sc_flush_request)
+        if (sb_rdonly(sb) || unlikely(!sci) || !sci->sc_flush_request)
                 return;

         set_bit(NILFS_SC_PRIOR_FLUSH, &sci->sc_flags);
@@ -2243,7 +2243,7 @@ int nilfs_construct_segment(struct super_block *sb)
         struct nilfs_transaction_info *ti;
         int err;

-        if (!sci)
+        if (sb_rdonly(sb) || unlikely(!sci))
                 return -EROFS;

         /* A call inside transactions causes a deadlock. */
@@ -2282,7 +2282,7 @@ int nilfs_construct_dsync_segment(struct super_block *sb, struct inode *inode,
         struct nilfs_transaction_info ti;
         int err = 0;

-        if (!sci)
+        if (sb_rdonly(sb) || unlikely(!sci))
                 return -EROFS;

         nilfs_transaction_lock(sb, &ti, 0);
@@ -2778,11 +2778,12 @@ int nilfs_attach_log_writer(struct super_block *sb, struct nilfs_root *root)

         if (nilfs->ns_writer) {
                 /*
-                 * This happens if the filesystem was remounted
-                 * read/write after nilfs_error degenerated it into a
-                 * read-only mount.
+                 * This happens if the filesystem is made read-only by
+                 * __nilfs_error or nilfs_remount and then remounted
+                 * read/write.  In these cases, reuse the existing
+                 * writer.
                  */
-                nilfs_detach_log_writer(sb);
+                return 0;
         }

         nilfs->ns_writer = nilfs_segctor_new(sb, root);

diff --git a/fs/nilfs2/super.c b/fs/nilfs2/super.c
index b5bc9f0c6a4063..049768a22388e3 100644
```

```
--- a/fs/nilfs2/super.c
+++ b/fs/nilfs2/super.c
@@ -1131,8 +1131,6 @@ static int nilfs_remount(struct super_block *sb, int *flags, char *data)
        if ((bool)(*flags & SB_RDONLY) == sb_rdonly(sb))
                goto out;
        if (*flags & SB_RDONLY) {
-               /* Shutting down log writer */
-               nilfs_detach_log_writer(sb);
                sb->s_flags |= SB_RDONLY;

                /*
```