author	Jan Kara <jack@suse.cz>	2022-01-17 18:22:13 +0100
committer	Jan Kara <jack@suse.cz>	2022-01-24 14:45:02 +0100
commit	7fc3b7c2981bbd1047916ade327beccb90994eee (patch)	
tree	410cae0a02add6911b15af81d8472738e50d0d8b	
parent	dd81e1c7d5fb126e5fbc5c9e334d7b3ec29a16a0 (diff)	
download	linux-7fc3b7c2981bbd1047916ade327beccb90994eee.tar.gz	

diff options

context:	3
space:	include
mode:	unified

udf: Fix NULL ptr deref when converting from inline format

```
udf_expand_file_adinicb() calls directly ->writepage to write data
expanded into a page. This however misses to setup inode for writeback
properly and so we can crash on inode->i_wb dereference when submitting
page for IO like:

  BUG: kernel NULL pointer dereference, address: 0000000000000158
  #PF: supervisor read access in kernel mode
...
  <TASK>
  __folio_start_writeback+0x2ac/0x350
  __block_write_full_page+0x37d/0x490
  udf_expand_file_adinicb+0x255/0x400 [udf]
  udf_file_write_iter+0xbe/0x1b0 [udf]
  new_sync_write+0x125/0x1c0
  vfs_write+0x28e/0x400

Fix the problem by marking the page dirty and going through the standard
writeback path to write the page. Strictly speaking we would not even
have to write the page but we want to catch e.g. ENOSPC errors early.

Reported-by: butt3rflyh4ck <butterflyhuangxx@gmail.com>
CC: stable@vger.kernel.org
Fixes: 52ebea749aae ("writeback: make backing_dev_info host cgroup-specific bdi_writebacks")
Reviewed-by: Christoph Hellwig <hch@lst.de>
Signed-off-by: Jan Kara <jack@suse.cz>
```

Diffstat

-rw-r--r--	fs/udf/inode.c	8

1 files changed, 3 insertions, 5 deletions

```
diff --git a/fs/udf/inode.c b/fs/udf/inode.c
index 1d6b7a50736ba..d6aa506b6b584 100644
--- a/fs/udf/inode.c
+++ b/fs/udf/inode.c
@@ -258,10 +258,6 @@ int udf_expand_file_adinicb(struct inode *inode)
 	char *kaddr;
 	struct udf_inode_info *iinfo = UDF_I(inode);
 	int err;
-	struct writeback_control udf_wbc = {
-		.sync_mode = WB_SYNC_NONE,
-		.nr_to_write = 1,
-	};
 
 	WARN_ON_ONCE(!inode_is_locked(inode));
```

```
	if (!iinfo->i_lenAlloc) {
@@ -305,8 +301,10 @@ int udf_expand_file_adinicb(struct inode *inode)
		iinfo->i_alloc_type = ICBTAG_FLAG_AD_LONG;
	/* from now on we have normal address_space methods */
	inode->i_data.a_ops = &udf_aops;
+	set_page_dirty(page);
+	unlock_page(page);
	up_write(&iinfo->i_data_sem);
-	err = inode->i_data.a_ops->writepage(page, &udf_wbc);
+	err = filemap_fdatawrite(inode->i_mapping);
	if (err) {
		/* Restore everything back so that we don't lose data... */
		lock_page(page);
```