author	Andrew Price <anprice@redhat.com>	2022-08-17 13:22:00 +0100
committer	Andreas Gruenbacher <agruenba@redhat.com>	2022-09-20 14:12:14 +0200
commit	670f8ce56dd0632dc29a0322e188cc73ce3c6b92 (patch)	
tree	e1d5fa41ed4cdd78445964dac6c8c0f83712cd65	
parent	204c0300c4e99707e9fb6e57840aa1127060e63f (diff)	
download	linux-670f8ce56dd0632dc29a0322e188cc73ce3c6b92.tar.gz	

gfs2: Check sb_bsize_shift after reading superblock

```
Fuzzers like to scribble over sb_bsize_shift but in reality it's very
unlikely that this field would be corrupted on its own. Nevertheless it
should be checked to avoid the possibility of messy mount errors due to
bad calculations. It's always a fixed value based on the block size so
we can just check that it's the expected value.

Tested with:

    mkfs.gfs2 -O -p lock_nolock /dev/vdb
    for i in 0 -1 64 65 32 33; do
        gfs2_edit -p sb field sb_bsize_shift $i /dev/vdb
        mount /dev/vdb /mnt/test && umount /mnt/test
    done

Before this patch we get a withdraw after

[   76.413681] gfs2: fsid=loop0.0: fatal: invalid metadata block
[   76.413681]   bh = 19 (type: exp=5, found=4)
[   76.413681]   function = gfs2_meta_buffer, file = fs/gfs2/meta_io.c, line = 492

and with UBSAN configured we also get complaints like

[   76.373395] UBSAN: shift-out-of-bounds in fs/gfs2/ops_fstype.c:295:19
[   76.373815] shift exponent 4294967287 is too large for 64-bit type 'long unsigned int'

After the patch, these complaints don't appear, mount fails immediately
and we get an explanation in dmesg.

Reported-by: syzbot+dcf33a7aae997956fe06@syzkaller.appspotmail.com
Signed-off-by: Andrew Price <anprice@redhat.com>
Signed-off-by: Andreas Gruenbacher <agruenba@redhat.com>
```

Diffstat

-rw-r--r-- fs/gfs2/ops_fstype.c 5

1 files changed, 4 insertions, 1 deletions

```
diff --git a/fs/gfs2/ops_fstype.c b/fs/gfs2/ops_fstype.c
index 236b59ef93b687..c7e2e623836685 100644
--- a/fs/gfs2/ops_fstype.c
+++ b/fs/gfs2/ops_fstype.c
@@ -178,7 +178,10 @@ static int gfs2_check_sb(struct gfs2_sbd *sdp, int silent)
```

```
                pr_warn("Invalid block size\n");
                return -EINVAL;
        }
-
+       if (sb->sb_bsize_shift != ffs(sb->sb_bsize) - 1) {
+               pr_warn("Invalid block size shift\n");
+               return -EINVAL;
+       }
        return 0;
 }
```