Subject: Re: [PATCH] remove BUG() in possible but rare condition
Posted by Glauber Costa on Wed, 11 Apr 2012 18:59:35 GMT
View Forum Message <> Reply to Message

On 04/11/2012 03:48 PM, Michal Hocko wrote:

```
> On Wed 11-04-12 15:10:24, Glauber Costa wrote:
>> While stressing the kernel with with failing allocations today,
>> I hit the following chain of events:
>>
>> alloc_page_buffers():
>>
>>  bh = alloc_buffer_head(GFP_NOFS);
>>  if (!bh)
>>   goto no_grow;<= path taken
>>
>> grow_dev_page():
>>         bh = alloc_page_buffers(page, size, 0);
>>         if (!bh)
>>                 goto failed;<= taken, consequence of the above
>>
>> and then the failed path BUG()s the kernel.
>>
>> The failure is inserted a litte bit artificially, but even then,
>> I see no reason why it should be deemed impossible in a real box.
>>
>> Even though this is not a condition that we expect to see
>> around every time, failed allocations are expected to be handled,
>> and BUG() sounds just too much. As a matter of fact, grow_dev_page()
>> can return NULL just fine in other circumstances, so I propose we just
>> remove it, then.
>
> I am not familiar with the code much but a trivial call chain walk up to
> write_dev_supers (in btrfs) shows that we do not check for the return value
> from __getblk so we would nullptr and there might be more.
> I guess these need some treat before the BUG might be removed, right?
```

You might very well be right, but if this is the case, this function is
probably wrong already.

find_or_create_page() failing will make it return NULL as well, and that
won't trigger the BUG() path.

At least in ext4 in my test case, the filesystem seems consistent after
a couple of runs
triggering this