Subject: [PATCH] remove BUG() in possible but rare condition
Posted by Glauber Costa on Wed, 11 Apr 2012 18:10:24 GMT
View Forum Message <> Reply to Message

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.

Signed-off-by: Glauber Costa <glommer@parallels.com>
CC: Linus Torvalds <torvalds@linux-foundation.org>
CC: Andrew Morton <akpm@linux-foundation.org>

```
 fs/buffer.c |    1 -
 1 files changed, 0 insertions(+), 1 deletions(-)

diff --git a/fs/buffer.c b/fs/buffer.c
index 36d6665..351e18e 100644
--- a/fs/buffer.c
+++ b/fs/buffer.c
@@ -985,7 +985,6 @@ grow_dev_page(struct block_device *bdev, sector_t block,
  return page;

 failed:
- BUG();
  unlock_page(page);
  page_cache_release(page);
  return NULL;
```

--
1.7.7.6

Subject: Re: [PATCH] remove BUG() in possible but rare condition
Posted by Michal Hocko on Wed, 11 Apr 2012 18:48:45 GMT
View Forum Message <> Reply to Message

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?

>
> Signed-off-by: Glauber Costa <glommer@parallels.com>
> CC: Linus Torvalds <torvalds@linux-foundation.org>
> CC: Andrew Morton <akpm@linux-foundation.org>
> ---
> fs/buffer.c | 1 -
> 1 files changed, 0 insertions(+), 1 deletions(-)
>
> diff --git a/fs/buffer.c b/fs/buffer.c

> index 36d6665..351e18e 100644
> --- a/fs/buffer.c
> +++ b/fs/buffer.c
> @@ -985,7 +985,6 @@ grow_dev_page(struct block_device *bdev, sector_t block,
> return page;
>
> failed:
> - BUG();
> unlock_page(page);
> page_cache_release(page);
> return NULL;
> --
> 1.7.7.6
>
> --
> To unsubscribe, send a message with 'unsubscribe linux-mm' in
> the body to majordomo@kvack.org. For more info on Linux MM,
> see: http://www.linux-mm.org/ .
> Fight unfair telecom internet charges in Canada: sign http://stopthemeter.ca/
> Don't email: email@kvack.org

--
Michal Hocko
SUSE Labs
SUSE LINUX s.r.o.
Lihovarska 1060/12
190 00 Praha 9
Czech Republic

Subject: Re: [PATCH] remove BUG() in possible but rare condition
Posted by Linus Torvalds on Wed, 11 Apr 2012 18:57:56 GMT
View Forum Message <> Reply to Message

On Wed, Apr 11, 2012 at 11:48 AM, Michal Hocko <mhocko@suse.cz> wrote:
>
> 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?

Well, realistically, isn't BUG() as bad as a NULL pointer dereference?

Do you care about the exact message on the screen when your machine dies?

Linus

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

Subject: Re: [PATCH] remove BUG() in possible but rare condition

Posted by Glauber Costa on Wed, 11 Apr 2012 19:02:19 GMT
View Forum Message <> Reply to Message

On 04/11/2012 03:57 PM, Linus Torvalds wrote:
> On Wed, Apr 11, 2012 at 11:48 AM, Michal Hocko<mhocko@suse.cz> wrote:
>>
>> 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?
>
> Well, realistically, isn't BUG() as bad as a NULL pointer dereference?
>
> Do you care about the exact message on the screen when your machine dies?
Not particular, but I don't see why (I might be wrong) it would
necessarily lead to a NULL pointer dereference.

At least in my test cases, after turning this to a WARN (to make sure it
was still being hit), the machine could go on just fine.

I was running this in a container system, with restricted memory. After
killing the container - at least in my ext4 system - everything seemed
as happy as ever.

Subject: Re: [PATCH] remove BUG() in possible but rare condition
Posted by Michal Hocko on Wed, 11 Apr 2012 19:20:52 GMT
View Forum Message <> Reply to Message

On Wed 11-04-12 11:57:56, Linus Torvalds wrote:
> On Wed, Apr 11, 2012 at 11:48 AM, Michal Hocko <mhocko@suse.cz> wrote:
> >
> > 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?
>
> Well, realistically, isn't BUG() as bad as a NULL pointer dereference?
>
> Do you care about the exact message on the screen when your machine dies?

I personally do not care as I do not allow anything to map at that area.

It just seems that there are some callers who do not expect that the
allocation fails. BUG at the allocation failure which dates back when it
replaced buffer_error might have let to some assumptions (not good of
course but we should better fix them.

That being said I am not against the patch. BUG on an allocation failure
just doesn't feel right...

>
>   Linus
>
> --
> To unsubscribe, send a message with 'unsubscribe linux-mm' in
> the body to majordomo@kvack.org. For more info on Linux MM,
> see: http://www.linux-mm.org/ .
> Fight unfair telecom internet charges in Canada: sign http://stopthemeter.ca/
> Don't email: email@kvack.org

--
Michal Hocko
SUSE Labs
SUSE LINUX s.r.o.
Lihovarska 1060/12
190 00 Praha 9
Czech Republic

Subject: Re: [PATCH] remove BUG() in possible but rare condition
Posted by Michal Hocko on Wed, 11 Apr 2012 19:25:54 GMT
View Forum Message <> Reply to Message

On Wed 11-04-12 16:02:19, Glauber Costa wrote:
> On 04/11/2012 03:57 PM, Linus Torvalds wrote:
> >On Wed, Apr 11, 2012 at 11:48 AM, Michal Hocko<mhocko@suse.cz> wrote:
> >>
> >>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?
> >
> >Well, realistically, isn't BUG() as bad as a NULL pointer dereference?
> >
> >Do you care about the exact message on the screen when your machine dies?
> Not particular, but I don't see why (I might be wrong) it would
> necessarily lead to a NULL pointer dereference.

Ahh, OK scratch that. I have misread __getblk_slow which returns NULL
only if grow_buffers returned with < 0 which doesn't happen for the
allocation failure.

Sorry about noise
--
Michal Hocko

SUSE Labs
SUSE LINUX s.r.o.
Lihovarska 1060/12
190 00 Praha 9
Czech Republic

Subject: Re: [PATCH] remove BUG() in possible but rare condition
Posted by akpm on Wed, 11 Apr 2012 20:26:35 GMT
View Forum Message <> Reply to Message

On Wed, 11 Apr 2012 15:10:24 -0300
Glauber Costa <glommer@parallels.com> 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.
>
> Signed-off-by: Glauber Costa <glommer@parallels.com>
> CC: Linus Torvalds <torvalds@linux-foundation.org>
> CC: Andrew Morton <akpm@linux-foundation.org>
> ---
> fs/buffer.c | 1 -
> 1 files changed, 0 insertions(+), 1 deletions(-)
>
> diff --git a/fs/buffer.c b/fs/buffer.c
> index 36d6665..351e18e 100644

> --- a/fs/buffer.c
> +++ b/fs/buffer.c
> @@ -985,7 +985,6 @@ grow_dev_page(struct block_device *bdev, sector_t block,
> return page;
>
> failed:
> - BUG();
> unlock_page(page);
> page_cache_release(page);
> return NULL;

Cute.

AFAICT what happened was that in my April 2002 rewrite of this code I
put a non-fatal buffer_error() warning in that case to tell us that
something bad happened.

Years later we removed the temporary buffer_error() and mistakenly
replaced that warning with a BUG(). Only it *can* happen.

We can remove the BUG() and fix up callers, or we can pass retry=1 into
alloc_page_buffers(), so grow_dev_page() "cannot fail". Immortal
functions are a silly fiction, so we should remove the BUG() and fix up
callers.

Subject: Re: [PATCH] remove BUG() in possible but rare condition
Posted by Glauber Costa on Wed, 11 Apr 2012 20:51:57 GMT
View Forum Message <> Reply to Message

On 04/11/2012 05:26 PM, Andrew Morton wrote:
>>
>> > failed:
>> > - BUG();
>> > unlock_page(page);
>> > page_cache_release(page);
>> > return NULL;
> Cute.
>
> AFAICT what happened was that in my April 2002 rewrite of this code I
> put a non-fatal buffer_error() warning in that case to tell us that
> something bad happened.
>
> Years later we removed the temporary buffer_error() and mistakenly
> replaced that warning with a BUG(). Only it*can* happen.
>
> We can remove the BUG() and fix up callers, or we can pass retry=1 into
> alloc_page_buffers(), so grow_dev_page() "cannot fail". Immortal

> functions are a silly fiction, so we should remove the BUG() and fix up
> callers.
>
Any particular caller you are concerned with ?

As I mentioned, this function already returns NULL for other reason -
that seem even more probable than this specific failure. So whoever is
not checking this return value, is already broken without this patch as
well.

Subject: Re: [PATCH] remove BUG() in possible but rare condition
Posted by akpm on Wed, 11 Apr 2012 21:12:44 GMT
View Forum Message <> Reply to Message

On Wed, 11 Apr 2012 17:51:57 -0300
Glauber Costa <glommer@parallels.com> wrote:

> On 04/11/2012 05:26 PM, Andrew Morton wrote:
> >>
> >> >   failed:
> >> > - BUG();
> >> >   unlock_page(page);
> >> >   page_cache_release(page);
> >> >   return NULL;
> > Cute.
> >
> > AFAICT what happened was that in my April 2002 rewrite of this code I
> > put a non-fatal buffer_error() warning in that case to tell us that
> > something bad happened.
> >
> > Years later we removed the temporary buffer_error() and mistakenly
> > replaced that warning with a BUG(). Only it*can* happen.
> >
> > We can remove the BUG() and fix up callers, or we can pass retry=1 into
> > alloc_page_buffers(), so grow_dev_page() "cannot fail". Immortal
> > functions are a silly fiction, so we should remove the BUG() and fix up
> > callers.
> >
> Any particular caller you are concerned with ?

Didn't someone see a buggy caller in btrfs?

I'm thinking that we should retain some sort of assertion (a WARN_ON)
if the try_to_free_buffers() failed. This is a weird case which I
assume handles the situation where a blockdev's blocksize has changed.
The code tries to throw away the old wrongly-sized buffer_heads and to
then add new correctly-sized ones. If that discarding of buffers

fails then the kernel is in rather a mess.

It's quite possible that this code is never executed - we _should_ have
invalidated all the pagecache for that device when changing blocksize.
Or maybe it *is* executed, I dunno. It's one of those things which has
hung around for decades as code in other places has vastly changed.

Subject: Re: [PATCH] remove BUG() in possible but rare condition
Posted by Michal Hocko on Wed, 11 Apr 2012 21:26:22 GMT
View Forum Message <> Reply to Message

On Wed 11-04-12 14:12:44, Andrew Morton wrote:
> On Wed, 11 Apr 2012 17:51:57 -0300
> Glauber Costa <glommer@parallels.com> wrote:
>
> > On 04/11/2012 05:26 PM, Andrew Morton wrote:
> > >>
> > >> > failed:
> > >> > - BUG();
> > >> > unlock_page(page);
> > >> > page_cache_release(page);
> > >> > return NULL;
> > > Cute.
> > >
> > > AFAICT what happened was that in my April 2002 rewrite of this code I
> > > put a non-fatal buffer_error() warning in that case to tell us that
> > > something bad happened.
> > >
> > > Years later we removed the temporary buffer_error() and mistakenly
> > > replaced that warning with a BUG(). Only it*can* happen.
> > >
> > > We can remove the BUG() and fix up callers, or we can pass retry=1 into
> > > alloc_page_buffers(), so grow_dev_page() "cannot fail". Immortal
> > > functions are a silly fiction, so we should remove the BUG() and fix up
> > > callers.
> > >
> > Any particular caller you are concerned with ?
>
> Didn't someone see a buggy caller in btrfs?

No I missed that __getblk (__getblk_slow) returns NULL only if
grow_buffers < 0 while it returns 0 for the allocation failure.

Sorry for confusion.
--
Michal Hocko
SUSE Labs

SUSE LINUX s.r.o.
Lihovarska 1060/12
190 00 Praha 9
Czech Republic

Subject: Re: [PATCH] remove BUG() in possible but rare condition
Posted by Jiri Kosina on Wed, 11 Apr 2012 21:33:07 GMT
View Forum Message <> Reply to Message

On Wed, 11 Apr 2012, Linus Torvalds wrote:

> > 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?
>
> Well, realistically, isn't BUG() as bad as a NULL pointer dereference?

Well, there still could be weirdos out there not setting
sys.vm.mmap_min_addr to something sane. For those, NULL pointer
dereference could have worse consequences than BUG (unlikely in this
particular case, yes).

--
Jiri Kosina
SUSE Labs

Subject: Re: [PATCH] remove BUG() in possible but rare condition
Posted by Michal Hocko on Thu, 12 Apr 2012 09:24:15 GMT
View Forum Message <> Reply to Message

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 had to be blind yesterday. I have double checked the call chain and
this is safe already because __getblk_slow tries to free some memory and
then retry the allocation if it gets allocation failure from
grow_buffers (grow_dev_page).

>
> Signed-off-by: Glauber Costa <glommer@parallels.com>
> CC: Linus Torvalds <torvalds@linux-foundation.org>
> CC: Andrew Morton <akpm@linux-foundation.org>

Reviewed-by: Michal Hocko <mhocko@suse.cz>

> ---
> fs/buffer.c | 1 -
> 1 files changed, 0 insertions(+), 1 deletions(-)
>
> diff --git a/fs/buffer.c b/fs/buffer.c
> index 36d6665..351e18e 100644
> --- a/fs/buffer.c
> +++ b/fs/buffer.c
> @@ -985,7 +985,6 @@ grow_dev_page(struct block_device *bdev, sector_t block,
>   return page;
>
> failed:
> - BUG();
>   unlock_page(page);
>   page_cache_release(page);
>   return NULL;
> --
> 1.7.7.6
>
> --
> To unsubscribe, send a message with 'unsubscribe linux-mm' in
> the body to majordomo@kvack.org. For more info on Linux MM,
> see: http://www.linux-mm.org/ .
> Fight unfair telecom internet charges in Canada: sign http://stopthemeter.ca/

> Don't email: email@kvack.org

--
Michal Hocko
SUSE Labs
SUSE LINUX s.r.o.
Lihovarska 1060/12
190 00 Praha 9
Czech Republic