

[PATCH] staging: lustre: add error handling for try\_module\_get

5 messages

Zhouyang Jia <jiazhouyang09@gmail.com>  
Cc: Zhouyang Jia <jiazhouyang09@gmail.com>, Oleg Drokin <oleg.drokin@intel.com>, Andreas Dilger <andreas.dilger@intel.com>, James Simmons <jsimmons@infradead.org>, Greg Kroah-Hartman <gregkh@linuxfoundation.org>, NeilBrown <neilb@suse.com>, Haneen Mohammed <hamohammed.sa@gmail.com>, Al Viro <viro@zeniv.linux.org.uk>, "Gustavo A. R. Silva" <garsilva@embeddedor.com>, lustre-devel@lists.lustre.org, devel@driverdev.osuosl.org, linux-kernel@vger.kernel.org

Tue, Jun 12, 2018 at 12:49 AM

When try\_module\_get fails, the lack of error-handling code may cause unexpected results.

This patch adds error-handling code after calling try\_module\_get.

Signed-off-by: Zhouyang Jia <jiazhouyang09@gmail.com>

```
---
drivers/staging/lustre/net/kinds/sockind/sockind.c | 5 +++++
1 file changed, 4 insertions(+), 1 deletion(-)

diff --git a/drivers/staging/lustre/net/kinds/sockind/sockind.c b/drivers/staging/lustre/net/kinds/sockind/sockind.c
index 7086678..72a42bd 100644
--- a/drivers/staging/lustre/net/kinds/sockind/sockind.c
+++ b/drivers/staging/lustre/net/kinds/sockind/sockind.c
@@@ -2422,7 +2422,10 @@@ ksocknal_base_startup(void)

    /* flag lists/ptrs/locks initialised */
    ksocknal_data.ksnd_init = SOCKNAL_INIT_DATA;
-   try_module_get(THIS_MODULE);
+   if (!try_module_get(THIS_MODULE)) {
+       CERROR("%s: cannot get module\n", __func__);
+       goto failed;
+   }

    ksocknal_data.ksnd_sched_info = cfs_percpt_alloc(net_cpt_table(),
                                                    sizeof(*info));
--
2.7.4
```

NeilBrown <neilb@suse.com>  
To: Zhouyang Jia <jiazhouyang09@gmail.com>  
Cc: Zhouyang Jia <jiazhouyang09@gmail.com>, Oleg Drokin <oleg.drokin@intel.com>, Andreas Dilger <andreas.dilger@intel.com>, James Simmons <jsimmons@infradead.org>, Greg Kroah-Hartman <gregkh@linuxfoundation.org>, Haneen Mohammed <hamohammed.sa@gmail.com>, Al Viro <viro@zeniv.linux.org.uk>, "Gustavo A. R. Silva" <garsilva@embeddedor.com>, lustre-devel@lists.lustre.org, devel@driverdev.osuosl.org, linux-kernel@vger.kernel.org


Tue, Jun 12, 2018 at 1:26 AM

On Tue, Jun 12 2018, Zhouyang Jia wrote:

```
> When try_module_get fails, the lack of error-handling code may
> cause unexpected results.
>
> This patch adds error-handling code after calling try_module_get.
>
> Signed-off-by: Zhouyang Jia <jiazhouyang09@gmail.com>
> ---
> drivers/staging/lustre/net/kinds/sockind/sockind.c | 5 +++++
> 1 file changed, 4 insertions(+), 1 deletion(-)
>
> diff --git a/drivers/staging/lustre/net/kinds/sockind/sockind.c b/drivers/staging/lustre/net/kinds/sockind/sockind.c
> index 7086678..72a42bd 100644
> --- a/drivers/staging/lustre/net/kinds/sockind/sockind.c
> +++ b/drivers/staging/lustre/net/kinds/sockind/sockind.c
> @@@ -2422,7 +2422,10 @@@ ksocknal_base_startup(void)
>
>     /* flag lists/ptrs/locks initialised */
>     ksocknal_data.ksnd_init = SOCKNAL_INIT_DATA;
> -   try_module_get(THIS_MODULE);
> +   if (!try_module_get(THIS_MODULE)) {
> +       CERROR("%s: cannot get module\n", __func__);
> +       goto failed;
> +   }
>
>     ksocknal_data.ksnd_sched_info = cfs_percpt_alloc(net_cpt_table(),
>                                                     sizeof(*info));
```

Thanks for the patch...  
I agree that this is probably a bug, but the code is still buggy after you patch, just in a different way.  
Try following through the code and see what happens when you 'goto failed'.

NeilBrown

 signature.asc  
1K

Greg Kroah-Hartman <gregkh@linuxfoundation.org>  
To: Zhouyang Jia <jiazhouyang09@gmail.com>  
Cc: Oleg Drokin <oleg.drokin@intel.com>, Andreas Dilger <andreas.dilger@intel.com>, James Simmons <jsimmons@infradead.org>, NeilBrown <neilb@suse.com>, Haneen Mohammed <hamohammed.sa@gmail.com>, Al Viro <viro@zeniv.linux.org.uk>, "Gustavo A. R. Silva" <garsilva@embeddedor.com>, lustre-devel@lists.lustre.org, devel@driverdev.osuosl.org, linux-kernel@vger.kernel.org

Tue, Jun 12, 2018 at 2:31 AM

```
On Tue, Jun 12, 2018 at 12:49:26PM +0800, Zhouyang Jia wrote:
> When try_module_get fails, the lack of error-handling code may
> cause unexpected results.
>
> This patch adds error-handling code after calling try_module_get.
>
> Signed-off-by: Zhouyang Jia <jiazhouyang09@gmail.com>
> ---
> drivers/staging/lustre/net/kinds/sockind/sockind.c | 5 +++++
```

This patch does not apply to Linus's tree. Always be sure to work against linux-next to catch things like this.

thanks,

greg k-h

David Laight <David.Laight@aculab.com>  
To: Zhouyang Jia <jiazhouyang09@gmail.com>  
Cc: Oleg Drokin <oleg.drokin@intel.com>, Andreas Dilger <andreas.dilger@intel.com>, James Simmons <jsimmons@infradead.org>, Greg Kroah-Hartman <gregkh@linuxfoundation.org>, NeilBrown <neilb@suse.com>, Haneen Mohammed <hamohammed.sa@gmail.com>, Al Viro <viro@zeniv.linux.org.uk>, "Gustavo A. R. Silva" <garsilva@embeddedor.com>, "lustre-devel@lists.lustre.org" <lustre-devel@lists.lustre.org>, "devel@driverdev.osuosl.org" <devel@driverdev.osuosl.org>, "linux-kernel@vger.kernel.org" <linux-kernel@vger.kernel.org>

Wed, Jun 13, 2018 at 6:53 AM

```
From: Zhouyang Jia
> Sent: 12 June 2018 05:49
>
> When try_module_get fails, the lack of error-handling code may
> cause unexpected results.
>
> This patch adds error-handling code after calling try_module_get.
...
> +++ b/drivers/staging/lustre/net/kinds/sockind/sockind.c
> @@@ -2422,7 +2422,10 @@@ ksocknal_base_startup(void)
>
>     /* flag lists/ptrs/locks initialised */
>     ksocknal_data.ksnd_init = SOCKNAL_INIT_DATA;
> -   try_module_get(THIS_MODULE);
> +   if (!try_module_get(THIS_MODULE)) {
> +       CERROR("%s: cannot get module\n", __func__);
> +       goto failed;
> +   }
> }
```

Can try\_module\_get(THIS\_MODULE) ever fail?  
Since you are running code in 'THIS\_MODULE' the caller must have a reference that can't go away.  
So try\_module\_get() just increments the count that is already greater than zero.

Similarly module\_put(THIS\_MODULE) must never be able to release the last reference.  
Any such calls that aren't in error paths after try\_module\_get() are probably buggy.

David

NeilBrown <neilb@suse.com>  
To: David Laight <David.Laight@aculab.com>, Zhouyang Jia <jiazhouyang09@gmail.com>  
Cc: Oleg Drokin <oleg.drokin@intel.com>, Andreas Dilger <andreas.dilger@intel.com>, James Simmons <jsimmons@infradead.org>, Greg Kroah-Hartman <gregkh@linuxfoundation.org>, Haneen Mohammed <hamohammed.sa@gmail.com>, Al Viro <viro@zeniv.linux.org.uk>, "Gustavo A. R. Silva" <garsilva@embeddedor.com>, "lustre-devel@lists.lustre.org" <lustre-devel@lists.lustre.org>, "devel@driverdev.osuosl.org" <devel@driverdev.osuosl.org>, "linux-kernel@vger.kernel.org" <linux-kernel@vger.kernel.org>

Wed, Jun 13, 2018 at 8:02 AM

On Wed, Jun 13 2018, David Laight wrote:

```
> From: Zhouyang Jia
>> Sent: 12 June 2018 05:49
>>
```

```

>> When try_module_get fails, the lack of error-handling code may
>> cause unexpected results.
>>
>> This patch adds error-handling code after calling try_module_get.
>
>> +++ b/drivers/staging/lustre/net/kinds/sockind/sockind.c
>> @@ -2422,7 +2422,10 @@ ksocknal_base_startup(void)
>>
>> /* flag lists/ptrs/locks initialised */
>> ksocknal_data.ksnd_init = SOCKNAL_INIT_DATA;
>> - try_module_get(THIS_MODULE);
>> + if (!try_module_get(THIS_MODULE)) {
>> +     CERROR("%s: cannot get module\n", __func__);
>> +     goto failed;
>> + }
>
> Can try_module_get(THIS_MODULE) ever fail?

```

Yes.

```

> Since you are running code in 'THIS_MODULE' the caller must have a
> reference that can't go away.

```

Not necessarily, though it does usually work that way.

try\_module\_get() can fail while the exit function is running, but it is safe to run code in the module until the exit function completes. So if the exit function takes a lock, then other code can safely run code in the module while holding the lock, but not holding a reference to the module. If this code calls try\_module\_get(), it could fail.

That is exactly what is happening here. ksocind\_exit() calls inet\_unregister\_ind() which takes the\_inetln\_ind\_mutex.

ksocknal\_base\_startup() is called from ksocknal\_startup() which is the\_ksockind.Ind\_startup and is called, from inet\_startup\_indni(), with that lock held.

```

> So try_module_get() just increments the count that is already greater
> than zero.
>
> Similarly module_put(THIS_MODULE) must never be able to release the
> last reference.

```

It can if a suitable lock is held.

```

> Any such calls that aren't in error paths after try_module_get() are
> probably buggy.


```

Being in an error path doesn't make it safe. module\_put(THIS\_MODULE) can only be safe if a lock is held which prevents the exit function from completing. Some code outside the module must release the lock.

Having said that, I don't really like this approach. I much prefer for the module reference to be taken and put outside of the module - it seems less error-prone.

NeilBrown

---

 **signature.asc**  
1K